

Strojové Učenie

Scikit-learn Regresia



Predictive Insights. Powered by AI

[Request Demo](#)

Miroslav Reiter

Kurz Strojové učenie v scikit-learn - Regresia

Vitajte v kurze strojového učenia zameraného na regresiu pomocou knižnice scikit-learn. Tento kurz vás prevedie základnými konceptmi strojového učenia, rôznymi typmi regresných modelov a ich praktickou implementáciou v Pythone. Naučíte sa, ako pripraviť dátá, trénovať modely a vyhodnocovať ich výkon.

Ako Začneme?

1. Pridajte si ma na LinkedIn

www.linkedin.com/in/miroslav-reiter

2. Mrknite na cvičenia

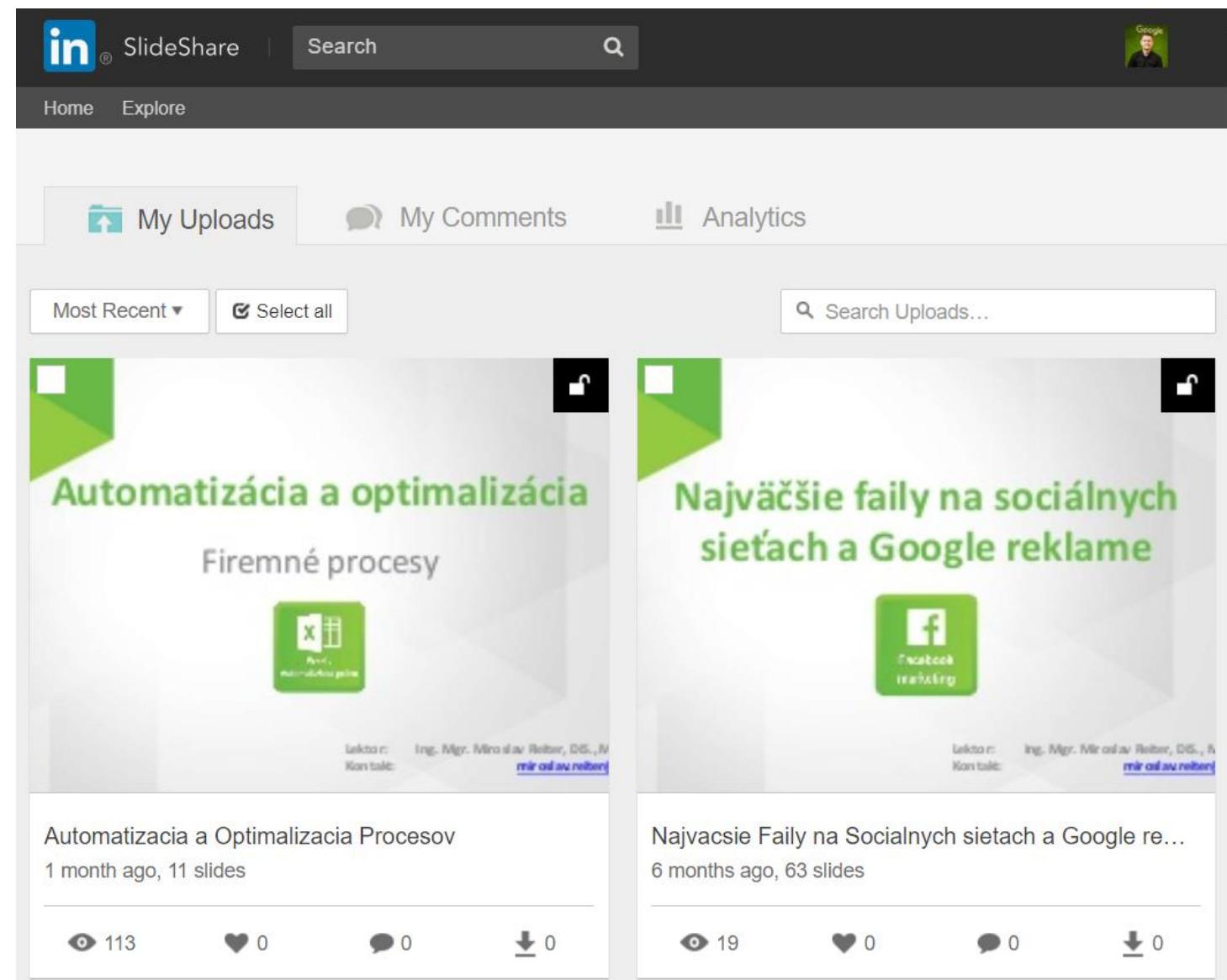
GitHub Repozitár

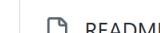
2. Prezentácia po prednáške

<https://github.com/miroslav-reiter/>

3. Videá na YouTube

www.youtube.com/@IT-Academy





🤖 Online kurz: Strojové učenie (Machine Learning) v Pythonе so scikit-learn

Praktický kurz pre začiatočníkov – Regresia v ML, scikit-learn, modelovanie, tréning a výhodnocovanie

Obsah kurzu

- 🔍 [Úvod do strojového učenia a regresie](#)
- 🧠 [Prehľad typov regresii](#)
- 📦 [Hlavné datasety v sklearn.datasets](#)
- 📈 [Lineárna regresia v scikit-learn](#)
- 📊 [Viacnásobná regresia a výber parametrov](#)
- 📚 [Zdroje a literatúra k strojovemu učeniu a scikit-learn](#)
- ✅ [Odporučania ML, regresia a Scikit-Learn](#)

1. Úvod do strojového učenia a regresie

🕒 Materiály k online kurzom a školeniam Strojové učenie (Machine Learning) v Pythonе so scikit-learn

🔗 www.vita.sk/online-kurz-scikit-learn/

python machine-learning scikit-learn

ml scikitlearn-machine-learning

strojove-ucenie

📄 Readme

📅 Activity

⭐ 0 stars

👁 0 watching

🍴 0 forks

Releases

No releases published

[Create a new release](#)

Packages

No packages published

[Publish your first package](#)



Kurzy DataScience, Python, R, Julia, BI, AI/ML, ChatGPT

Materiály, Zdrojové Kódy a Projekty, Prezentácie ku kurzom DataScience, Python, OOP, R, BI, Analytika

Python je interpretovaný, interaktívny, open-source programovací jazyk. Python beží na mnohých variantoch Unixu, na Macu a Windowse (súčasťou kurzu bude inštalácia na vašom systéme). Pre absolvovanie kurzu je potrebné mať k dispozícii vlastný notebook (s ľubovoľným operačným systémom podporujúcim Python).

R je programovací jazyk a softvérové prostredie pre štatistickú analýzu a vizualizáciu. R je voľne dostupný pod GNU licenciou a existujú verzie pre mnohé operačné systémy ako Linux, Windows a Mac.

ChatGPT je pokročilý jazykový model umejelj inteligencie vyvinutý spoločnosťou OpenAI. Je to variant modelu GPT (Generative Pre-trained Transformer), ktorá je špeciálne navrhnutá na generovanie textu a interakciu v dialógovej forme. Bol vyučovaný na veľkom množstve textových dát z internetu, čím získal schopnosť generovať koherentné a relevantné odpovede na zadanej otázky alebo pokyny v prirodzenom jazyku. ChatGPT, ako variant modelu GPT vyvinutého spoločnosťou OpenAI, sa primárne trénuje pomocou metódy zvanéj semi-supervised learning, ktorá je kombináciou supervised (riadenej) a unsupervised (neriadenej) učenia.

Používané nástroje

1 Jupyter Notebooks

About

Materiály, Zdrojové Kódy, Prezentácie ku kurzom Python, OOP, R, BI, Data Science, AI/ML, ChatGPT

www.vita.sk

python jupyter numpy oop
jupyter-notebook pandas python3
matplotlib sav beatifulsoup reiter
matplotlib-pyplot

Readme

Activity

23 stars

16 watching

25 forks

Releases

No releases published
[Create a new release](#)

Packages

No packages published
[Publish your first package](#)

Languages



Suggested workflows

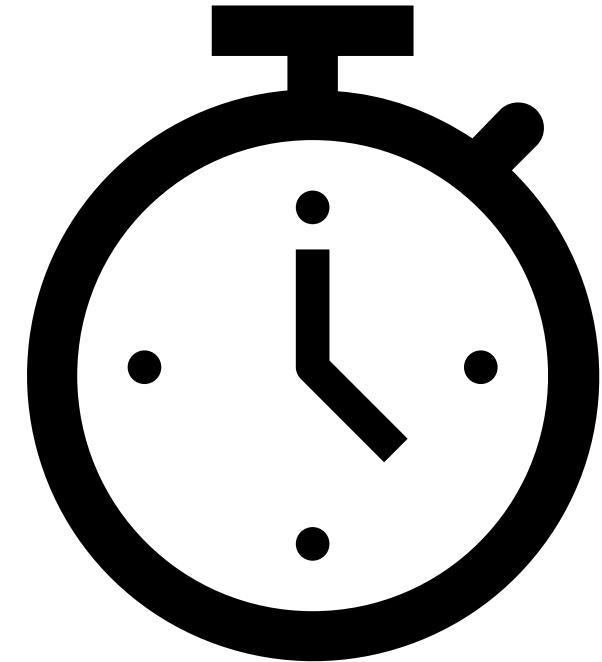
Based on your tech stack

Python application

Configure

Úvodné Informácie

- Časový rozvrh (9:00-13:30)
 - Pracujeme (50 min)
 - Prestávky (10 min)
 - Obedová prestávka
 - Mobilné telefóny a zariadenia
-
- Priprav si otázky a rovno sa pýtaj
 - Interaktívna forma



O mne - Miroslav Reiter

7

40000+
klientov a
1000+ firiem

IT Architekt
Programátor
Manažér

Microsoft
Google
ISTQB tréner

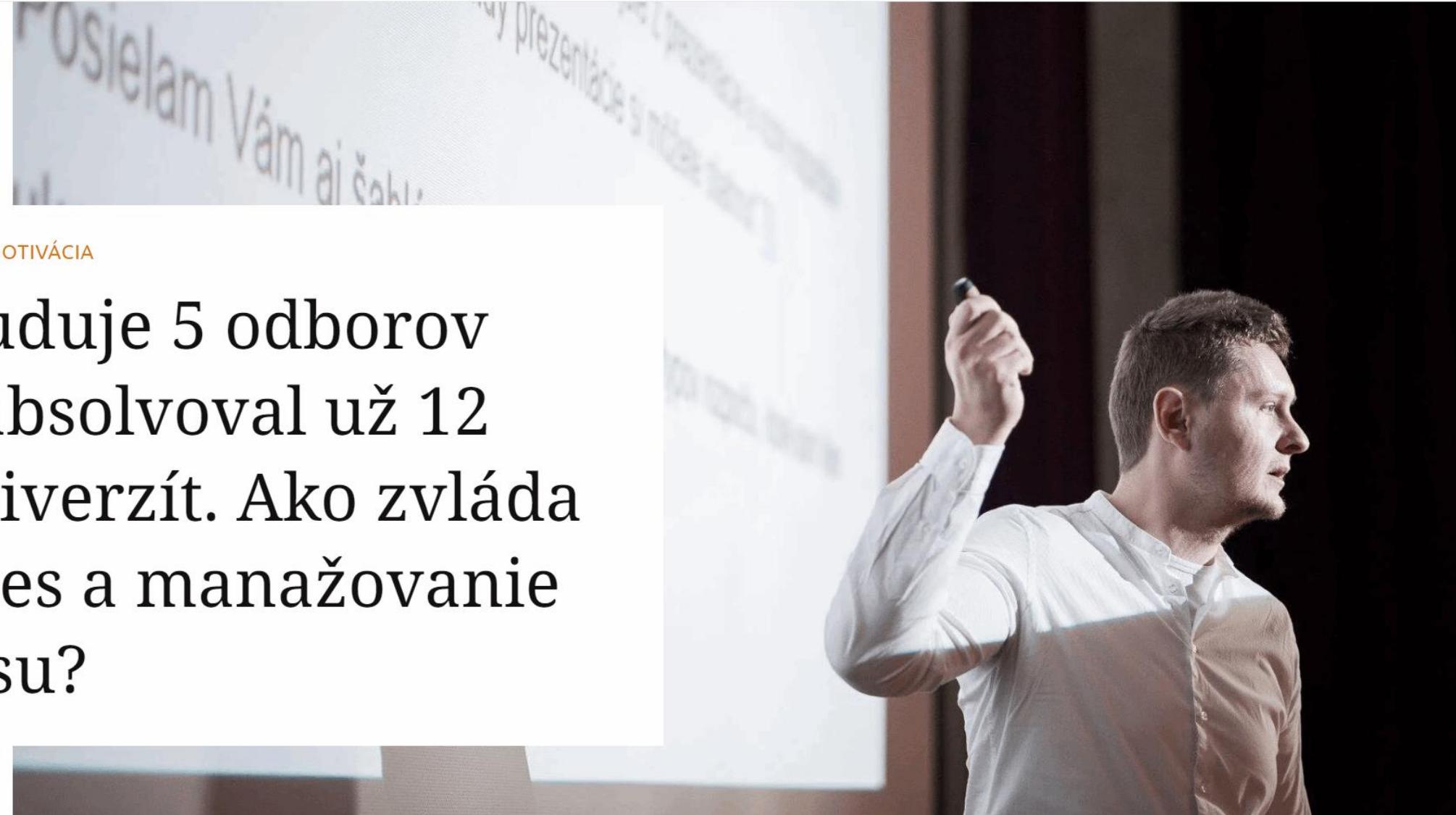
134 certifikácií

151 príručiek
a publikácií

13 škôl

62 projektov

Vlastná firma



MOTIVÁCIA

Študuje 5 odborov a absolvoval už 12 univerzít. Ako zvláda stres a manažovanie času?

Foto: Jakub Kovalík pre FMK UCM | Miroslav Reiter na prednáške Grow with Google na FMK UCM.



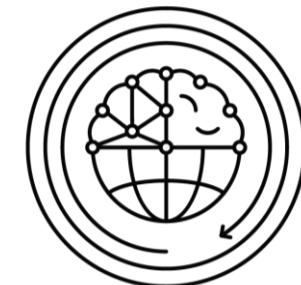
Nikola Kotláriková
19. júl 2022 · 8 min. čítania



Miroslav Reiter



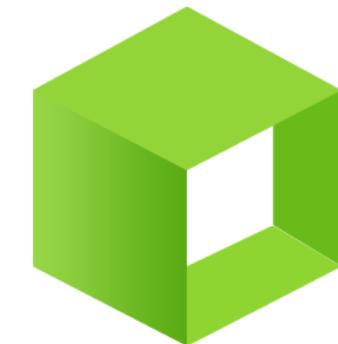
1. PhDr. VŠM (Podnikovný manažment)
2. Ing. STU FEI (**Aplikovaná informatika**)
3. Mgr. UK FM (**Strategický manažment a marketing**)
4. Mgr. VŠM (**Manažérstvo kvality**)
5. Mgr. VŠEMVŠ (Verejná správa)
6. Mgr. DTI (**Učiteľstvo ekonomických predmetov**)
7. DiS. AMOS (Cestovný ruch)
8. **MBA LIGS (Executive management)**
9. **DBA Humanum (IT manažment)**
10. MPA IES (Verejná správa a samospráva krajov)
11. MSc. Humanum (**Bezpečnosť inf. systémov**)
12. Ing. Paed. IGIP STU
13. Mgr. PEVŠ (**Bezpečnosť informačných systémov**)
14. RNDR. PEVŠ (**Bezpečnosť informačných systémov**)



FAKULTA MANAGEMENTU
Univerzita Komenského
v Bratislave

**DIGITÁLNA
UNIVERZITA**

STU
FIIT



IT ACADEMY

 **VITA**
ACADEMY

Most Valuable Professionals

[About](#) [Events](#) [Find an MVP](#)[Profile](#) [Events](#)

Headline

 IT Architect and Programmer  Hard worker 
Lecturer and Certified Trainer

Biography

 I'm a hard worker, intellectual and joker. I love learning and teaching as well. My main objective is to teach people and improve their IT knowledge. To create truly practical knowledge necessary for life and present it in an interesting way. I don't like snake charmers and people who cannot...

[▽ Read more](#)

Miroslav Reiter

 Slovakia IT Academy s.r.o.

Most Valuable Professionals

High Impact Activities

Award Category

M365

Technology Area

Visio, Excel

Languages

English, Slovak

Social



This community leader has not added a high impact activity yet.

Miroslav Reiter

získava status
Google Certified Trainer

Automation

Google



 [Domov](#) > [Registre](#) > [Znalcí](#) > PhDr. Ing. Miroslav Reiter

Znalec

PhDr. Ing. Mgr. Miroslav Reiter

Evidenčné číslo: 915864

Miesto výkonu činnosti

Tomášikova 50
83104 Bratislava
Slovenská republika

[Zobraziť na mape](#)

Kontaktné údaje

Mobil: +421 908 163 084
E-mail: znapec@it-academy.sk

Odbory a odvetvia

Odbor / Odvetvie	História	Stav
100000 - Elektrotechnika		
100400 - Riadiaca technika, výpočtová technika (hardware)	14.02.2023 - Zápis	AKTÍVNY
100800 - Nosiče zvukových a zvukovoobrazových záznamov	14.02.2023 - Zápis	AKTÍVNY
100900 - Počítačové programy (software)	14.02.2023 - Zápis	AKTÍVNY
101000 - Bezpečnosť a ochrana informačných systémov	14.02.2023 - Zápis	AKTÍVNY

Vyberte si online kurz

Naučte sa programovať, tvoriť webstránky a grafiku, manažovať alebo sa zamerajte na osobný rozvoj. Všetko jednoducho vďaka našim online kurzom z pohodlia domova.

Ročné Predplatné na všetky Online Kurzy

2200 €

490 €

Prístup pre Vás do všetkých Aktuálnych aj
Pripravovaných Online Kurzov

12 mesačná platnosť

Kúpiť teraz

Získať viac



573 kurzov v ponuke



Zábavné online lekcie



Akreditované kurzy



13 rokov skúseností



Certifikovaní profesionálni lektori

Odporúčame Kurzy špeciálne pre vás



Balík Digitálny Marketing a Eshop

925,60€ 925,60€



Ročné predplatné kategórie SAP a ABAP

390,00€ 590,00€



Online kurz ChatGPT a DALL-E AI I. Začiatočník

186,00€ 254,00€



Online kurz SAP I. Začiatočník

298,00€ 398,00€



Online kurz Projektový Manažér I. Začiatočník

198,00€ 286,00€



Program MSc SAP Špecialista (SAP Consultant)

1 940,00€ 2 540,00€

[Všetko](#) [Obrázky](#) [Videá](#) [Správy](#) [Krátke videá](#) [Knihy](#) [Web](#) [Viac ▾](#)[Nástroje ▾](#)

Online kurz Strojové Učenie sa zameriava na proces zisťovania a opravy (alebo odstránenia) nepresných záznamov zo sady záznamov, tabuľky alebo databázy a týka sa identifikácie neúplných, nesprávnych, nepresných alebo irelevantných častí údajov a následného nahradenia, úpravy, alebo vymazanie špinavých alebo hrubých ...

 VITA Academy<https://www.vita.sk> · Shop · Umelá Inteligencia AI

⋮

[Online kurz Strojové Učenie \(Machine Learning ML\) | VITA](#)

? Vybrané úryvky • ! Spätná väzba

Videá :

[Online kurz Machine Learning ML - Čo je Strojové Učenie ...](#)YouTube · Miroslav Reiter - VITA Academy
30. 9. 2024

3 kľúčové momenty v tomto videu

[Online kurz Machine Learning v Python - Klastrovanie ...](#)YouTube · Miroslav Reiter - VITA Academy
1. 11. 2024[Online Kurz Umelá inteligencia AI, Automatizácia, Strojové ...](#)YouTube · Miroslav Reiter - VITA Academy
13. 5. 2021

11 kľúčových momentov v tomto videu

Zobrazit všetky >

 VITA Academy<https://www.vita.sk> · Shop

⋮

[Kurzy Umelá Inteligencia a Strojové Učenie \(AI a ML\)](#)

[Všetko](#) [Obrázky](#) [Videá](#) [Krátke videá](#) [Správy](#) [Knihy](#) [Web](#) [Viac ▾](#)[Nástroje ▾](#)

www.youtube.com › watch

Online kurz Scikit-Learn Python - Prehľad Machine Learning ...



Absolvujte celý online kurz na → <https://www.vita.sk/online-kurz-scikit-learn/>

Online kurz knižnica Scikit-Learn v Python je pre vás ideálny ...

YouTube · Miroslav Reiter - VITA Academy · 25. 10. 2024

www.youtube.com › watch

Online kurz Dátové Čistenie a Predspracovanie Dát Python ...



Absolvujte celý **online kurz** na → [YouTube · Miroslav Reiter - VITA Academy · 18. 10. 2024](https://www.vita.sk/online-kurz... Online kurz Dátové Čistenie a Predspracovanie Dát Python Scikit-learn.</p></div><div data-bbox=)

www.youtube.com › watch

Online kurz Machine Learning ML - Čo je Strojové Učenie ...



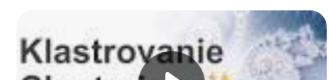
Online kurz Machine Learning v Python - Klastrovanie Zhlukovanie Dát (Clustering) ★ ... Machine Learning with Python and **Scikit-Learn** ...

YouTube · Miroslav Reiter - VITA Academy · 30. 9. 2024



www.youtube.com › watch

Online kurz Machine Learning v Python - Klastrovanie ...



... Kurz kladie dôraz na praktické využitie knižnice **Scikit-Learn**, ktorá je

kľúčovým nástrojom pre implementáciu algoritmov strojového učenia v ...

[Všetko](#) [Obrázky](#) [Videá](#) [Správy](#) [Knihy](#) [Finance](#) [Krátke videá](#) [Viac](#)[Nástroje](#)

Videá :

[Online kurz Štatistika - Čo je to Regresia a Regresná Analýza ...](#)

YouTube · Miroslav Reiter - VITA Academy

pred 3 týždňami

4 kľúčové momenty v tomto video

Zdroj: 00:00

Úvod

Zdroj: 10:00

Spearmanov
Korelačný
Koeficient

Zdroj: 20:00

Použitie
Chyby Korelácia

Koeficientov

Zdroj: 30:00

Chyby Korelácia

[UK FM - Kurz Štatistické Metódy h. SAS Enterprise Guide ...](#)

YouTube · Miroslav Reiter - VITA Academy

15. 4. 2021



10 kľúčových momentov v tomto video

[Online kurz Scikit-Learn Python - Prehľad Machine Learning ...](#)

YouTube · Miroslav Reiter - VITA Academy

25. 10. 2024

[UK FM - Kurz Štatistické Metódy e. SAS Enterprise Guide ...](#)

YouTube · Miroslav Reiter - VITA Academy

1. 4. 2021

[Zobraziť všetky >](#)

VITA Academy

<https://www.vita.sk> › online-kurzy-analyza-dat

Kurzy Analýza Dát

AKREDITOVANÉ VZDELÁVACIE PROGRAMY



Vyhľadávanie akreditovaných vzdelávacích programov
(vyplňte vstupné polia, podľa ktorých chcete vyhľadávať)

Názov programu / modulu

python

Číslo akreditácie / štvorčíslo**Mesto / okres / kraj****Vzdelávacia inštitúcia****Vstupné vzdelanie****Organizačná forma****Vyhľadať**[Jednoduché vyhľadávanie](#)**Počet nájdených záznamov: 15**

Názov	Dátum akreditácie	Rozsah	Inštitúcia	Mesto
Programovacie jazyky / Python I. - Základy_(POA: 3428/2020/122/5)	10. 12. 2020	10,0 hod.	IT Academy s. r. o.	Bratislava
Programovacie jazyky / Python II. - Mierne pokročilé techniky_(POA: 3428/2020/122/5)	10. 12. 2020	10,0 hod.	IT Academy s. r. o.	Bratislava
Programovacie jazyky / Python - Automatizácia Práce (POA: 3428/2020/122/5)	10. 12. 2020	10,0 hod.	IT Academy s. r. o.	Bratislava
Programovacie jazyky / Python - Knižnice_(POA: 3428/2020/122/5)	10. 12. 2020	10,0 hod.	IT Academy s. r. o.	Bratislava
Programovacie jazyky / Python - Pandas_(POA: 3428/2020/122/5)	10. 12. 2020	10,0 hod.	IT Academy s. r. o.	Bratislava
Programovacie jazyky / Python Django I. - Základy_(POA: 3428/2020/122/5)	10. 12. 2020	10,0 hod.	IT Academy s. r. o.	Bratislava
Programovacie jazyky / Python Django II. - Mierne pokročilé techniky_(POA: 3428/2020/122/5)	10. 12. 2020	10,0 hod.	IT Academy s. r. o.	Bratislava
Programovacie jazyky / Python Django III. - Pokročilé techniky_(POA: 3428/2020/122/5)	10. 12. 2020	10,0 hod.	IT Academy s. r. o.	Bratislava
Programovacie jazyky / Python III. - Pokročilé techniky_(POA: 3428/2020/122/5)	10. 12. 2020	10,0 hod.	IT Academy s. r. o.	Bratislava
Programovacie jazyky / Python IV. - Návrhové Vzory_(POA: 3428/2020/122/5)	10. 12. 2020	10,0 hod.	IT Academy s. r. o.	Bratislava

1 2

Adresa pracoviska:

Ministerstvo školstva, výskumu, vývoja a mládeže Slovenskej republiky

Sekcia stredných škôl a celoživotného vzdelávania

Odbor vzdelávania dospelých

Stromová 1

813 30 Bratislava 1

Online kurz Knižnica Scikit-Learn v Python

0 % už máš za sebou

The screenshot shows a video player interface. At the top, it says "Strojové Učenie ML - Knižnica Scikit-learn 2024". The main content features a large title "Knižnica Scikit-learn" in bold black font, accompanied by a green snake logo and a red book icon. Below the title is a stack of various colored books. The video player has a progress bar at the bottom showing 31:07. On the left, there's a sidebar with a list of video thumbnails and titles, such as "Obsah (1)", "Úvod do Strojoveh...", and "Knižnica Scikit-learn". A purple button at the bottom left says "Pokračovať na ďaľšiu časť >". The video player has a "vimeo" logo at the bottom right.

[Prehľad](#) [Materiály](#) [Otázky a odpovede](#) [Vyhľadávanie](#)

Materiály

- Materiály kurz Strojové Učenie (Odkaz vo vnútri súboru)

I. Úvod a predstavenie knižnice Scikit-Learn

Videné

[Knižnica Scikit-learn](#)

31:07

[Dátové čistenie a predspracovanie dát](#)

01:25:44

[Normalizácia reťazcov](#)

09:57

II. Jazyk Python, validácia a data parsing

Videné

[Validácia a normalizácia dát](#)

15:46

[Dátové čistenie v Pythone a reťazce](#)

1:42:03

[Reťazce, kolekcie, deque](#)

30:09

[Parsovanie dát](#)

52:40

[Parsovanie XML](#)

55:43

[Parsovanie URL, logy, PDF](#)

53:16

III. Klastrovanie (Zhlukovanie) Dát

Videné

[Čo je klastrovanie](#)

35:46

[Klastrovanie typy](#)

11:13

Chat



Luigi, Mário
a Yoshi

Čo Robíte?

1. Študent/učiteľ

2. Zamestnanec

3. Podnikateľ

4. Nezamestnaný/materská

5. Dievča pre všetko



Agenda kurzu

- 1 Úvod do ML a Regresie
- 2 Prehľad typov regresií
- 3 Lineárna regresia
- 4 Viacnásobná regresia



Vaše Ciele a Očakávania

1. Doplniť si znalosti z jazyka Python

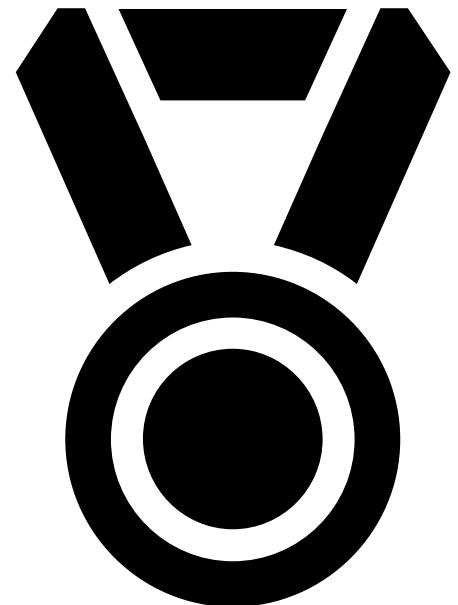
2. Úvod do regresie

3. Lineárna regresia

4. Viacnásobná regresia

5. Doplniť si znalosti z IT

Zábava je v zaručená v každom bode 😊



Ako začať so Strojovým Učením?

AKREDITOVANÝ KURZ





Čo sa naučíme?

1. Čo je strojové učenie (machine learning) a prečo sa používa?
2. Aké poznáme typy učenia?
3. Čo znamená supervised learning?
4. Čo je regresia v strojovom učení?
5. Aký je rozdiel medzi regresiou a klasifikáciou?
6. Na čo slúži knižnica scikit-learn?
7. Profit

Základné pojmy strojového učenia



Strojové učenie (ML)

Disciplína umelej inteligencie umožňujúca systémom učiť sa zo skúseností (dát) bez explicitného naprogramovania pravidiel.



Učenie s učiteľom

Model má k dispozícii vstupy a správne výstupy – cieľom je naučiť sa pravidlo ich prepojenia.



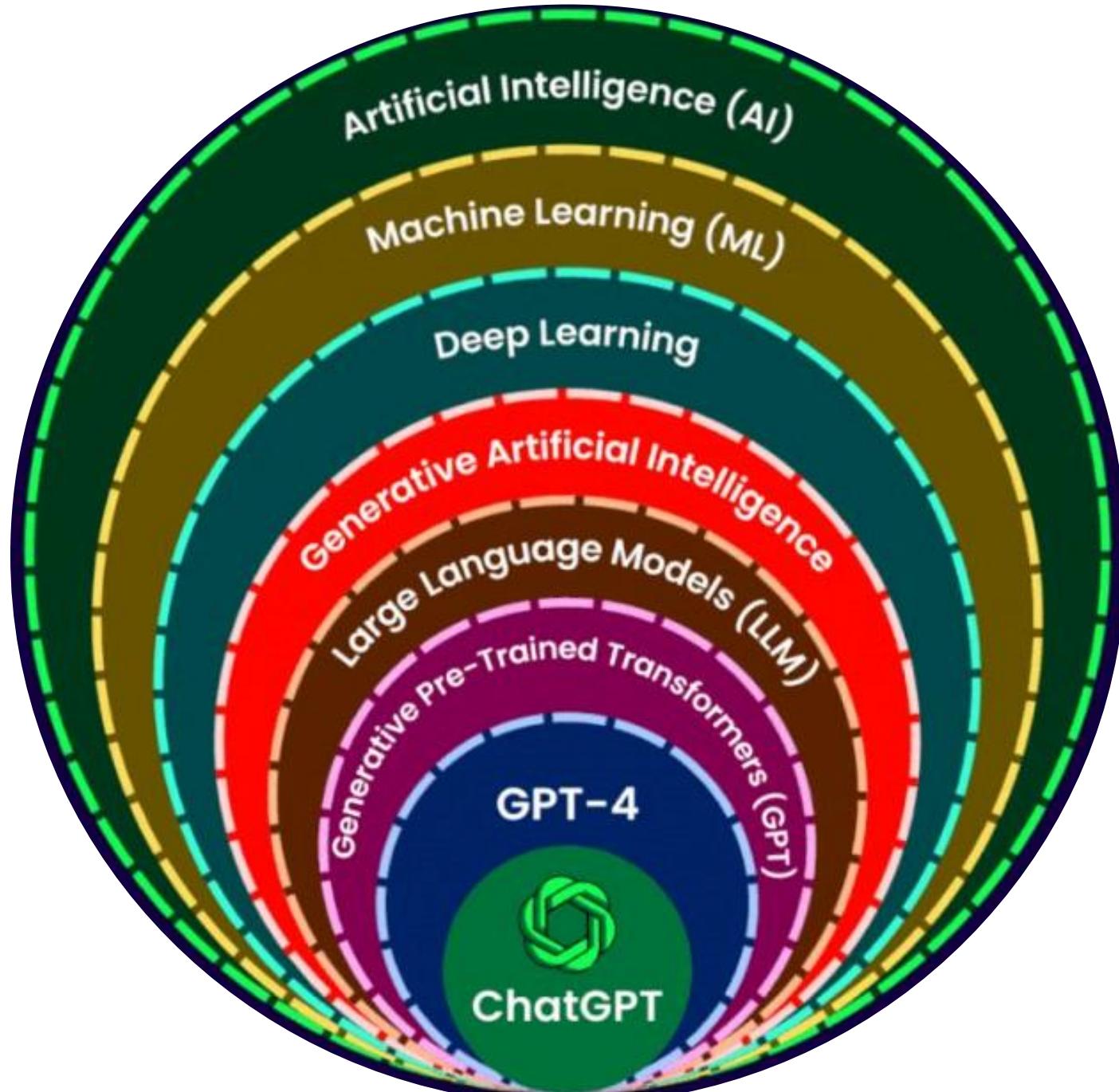
Učenie bez učiteľa

Model dostane iba vstupy – cieľom je nájsť vzory alebo štruktúru v dátach (napr. klastrovanie).

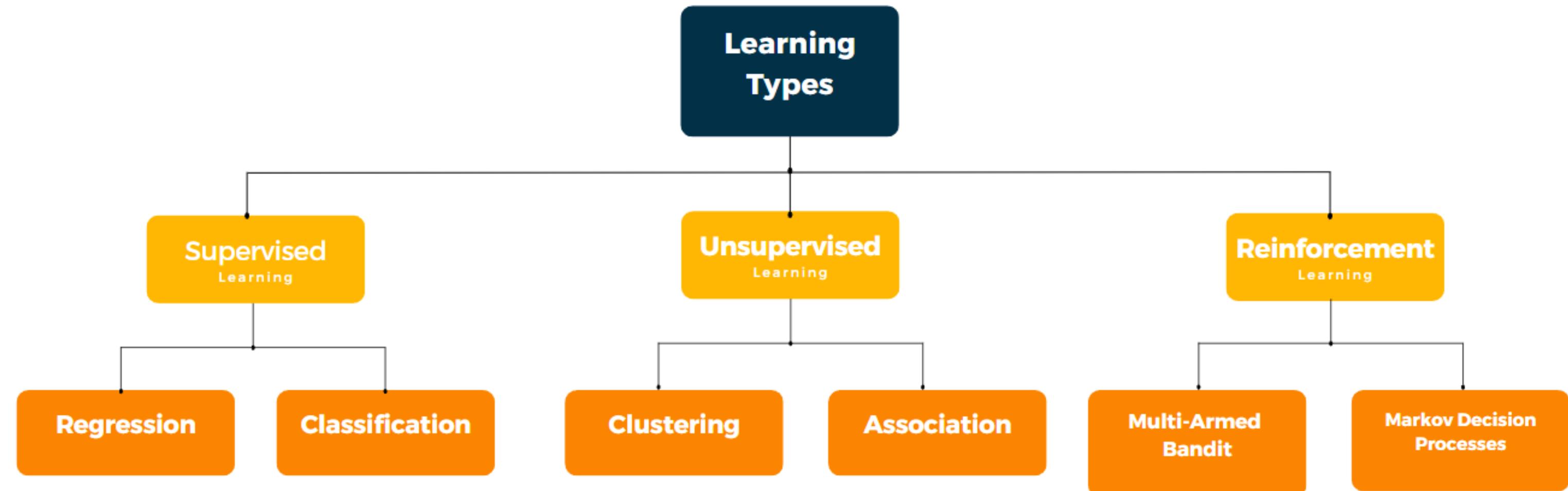


Čo je Machine Learning?

- Oblast' umelej inteligencie, ktorá sa zaoberá vývojom algoritmov a modelov, ktoré umožňujú počítačom učiť sa z údajov a zlepšovať svoje výkony bez explicitného programovania
- Modely sa trénujú na historických dátach, aby dokázali predpovedať alebo klasifikovať nové, neznáme údaje



Typy Strojového Učenia



Typy Strojového Učenia

A. Učenie s učiteľom (Supervised Learning)

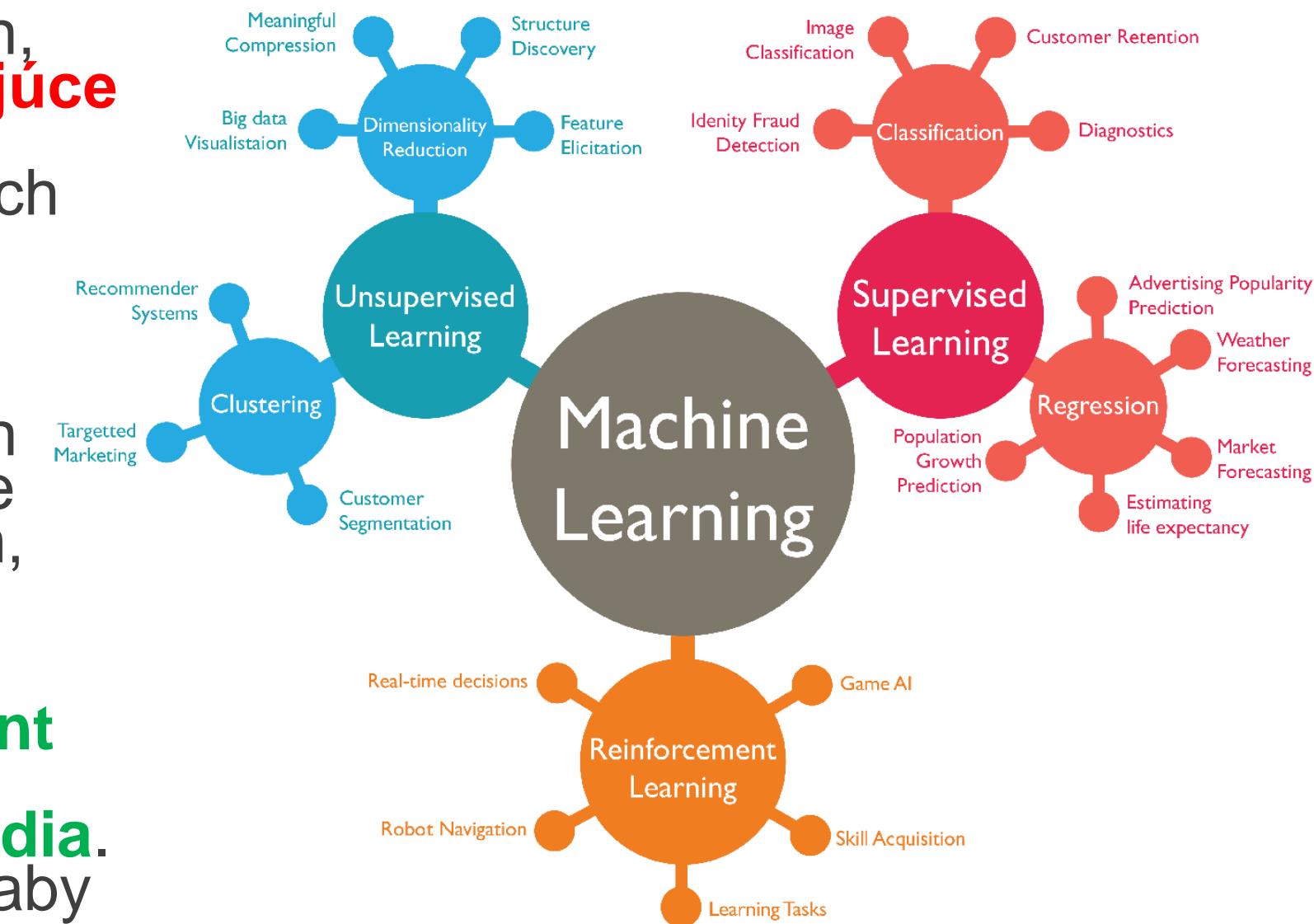
Model sa trénuje na dátach, ktoré **obsahujú vstupy a zodpovedajúce výstupy**. Cieľom je naučiť model predpovedať výstupy na základe nových vstupov. Príkladmi sú **regresia** a **klasifikácia**.

B. Učenie bez učiteľa (Unsupervised Learning)

Model sa trénuje na dátach **bez explicitných výstupov**. Cieľom je objaviť vzory alebo štruktúry v údajoch, ako je **zhlukovanie** alebo **redukcia dimenzií**.

C. Učenie so zosilnením (Reinforcement Learning)

Model sa učí na základe skúseností a **spätnej väzby z prostredia**. Cieľom je **optimalizovať sériu akcií**, aby sa dosiahol maximálny zisk.



Učenie s učiteľom



- **Regresia:**
Predpovedanie cien nehnuteľností
na základe faktorov ako plocha,
počet izieb a lokalita
- **Klasifikácia:**
Rozpoznávanie e-mailov ako spam
alebo nie spam na **základe obsahu**
a metadát

NETFLIX

 Zillow

facebook

Python Knižnice

Data Manipulation



Data Visualization



Statistical Analysis



Machine Learning



Database Operations



Time Series Analysis



Web Scraping



scikit-learn

Machine Learning in Python

[Getting Started](#)[Release Highlights for 1.6](#)

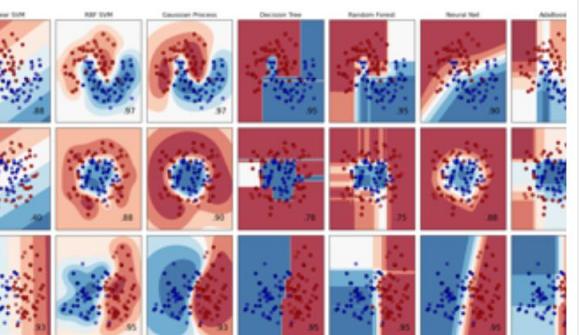
- Simple and efficient tools for predictive data analysis
- Accessible to everybody, and reusable in various contexts
- Built on NumPy, SciPy, and matplotlib
- Open source, commercially usable - BSD license

Classification

Identifying which category an object belongs to.

Applications: Spam detection, image recognition.

Algorithms: [Gradient boosting](#), [nearest neighbors](#), [random forest](#), [logistic regression](#), and [more...](#)

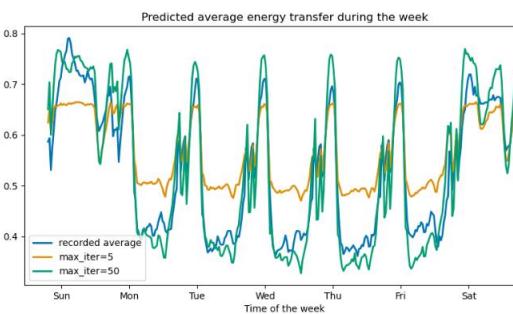
[Examples](#)

Regression

Predicting a continuous-valued attribute associated with an object.

Applications: Drug response, stock prices.

Algorithms: [Gradient boosting](#), [nearest neighbors](#), [random forest](#), [ridge](#) and [more...](#)

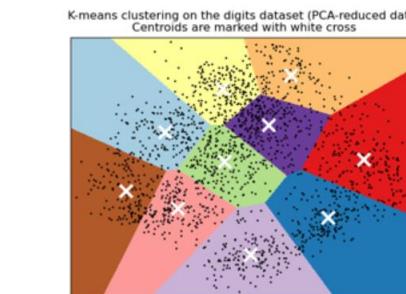
[Examples](#)

Clustering

Automatic grouping of similar objects into sets.

Applications: Customer segmentation, grouping experiment outcomes.

Algorithms: [k-Means](#), [HDBSCAN](#), [hierarchical clustering](#), and [more...](#)

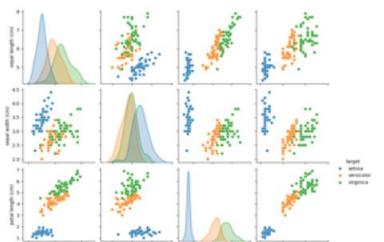
[Examples](#)

Dimensionality reduction

Reducing the number of random variables to consider.

Applications: Visualization, increased efficiency.

Algorithms: [PCA](#), [feature selection](#), [non-negative matrix factorization](#), and [more...](#)

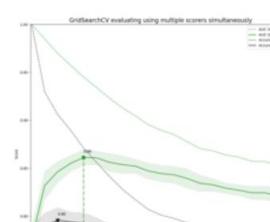


Model selection

Comparing, validating and choosing parameters and models.

Applications: Improved accuracy via parameter tuning.

Algorithms: [Grid search](#), [cross validation](#), [metrics](#), and [more...](#)

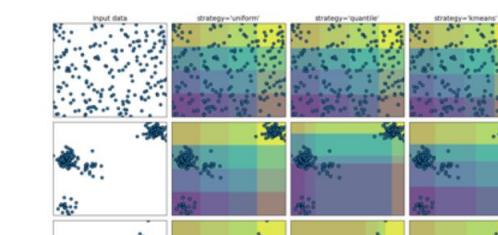


Preprocessing

Feature extraction and normalization.

Applications: Transforming input data such as text for use with machine learning algorithms.

Algorithms: [Preprocessing](#), [feature extraction](#), and [more...](#)





Použitie Skikit-learn

**1. Klasifikácia
(Classification)**

**2. Regresia
(Regression)**

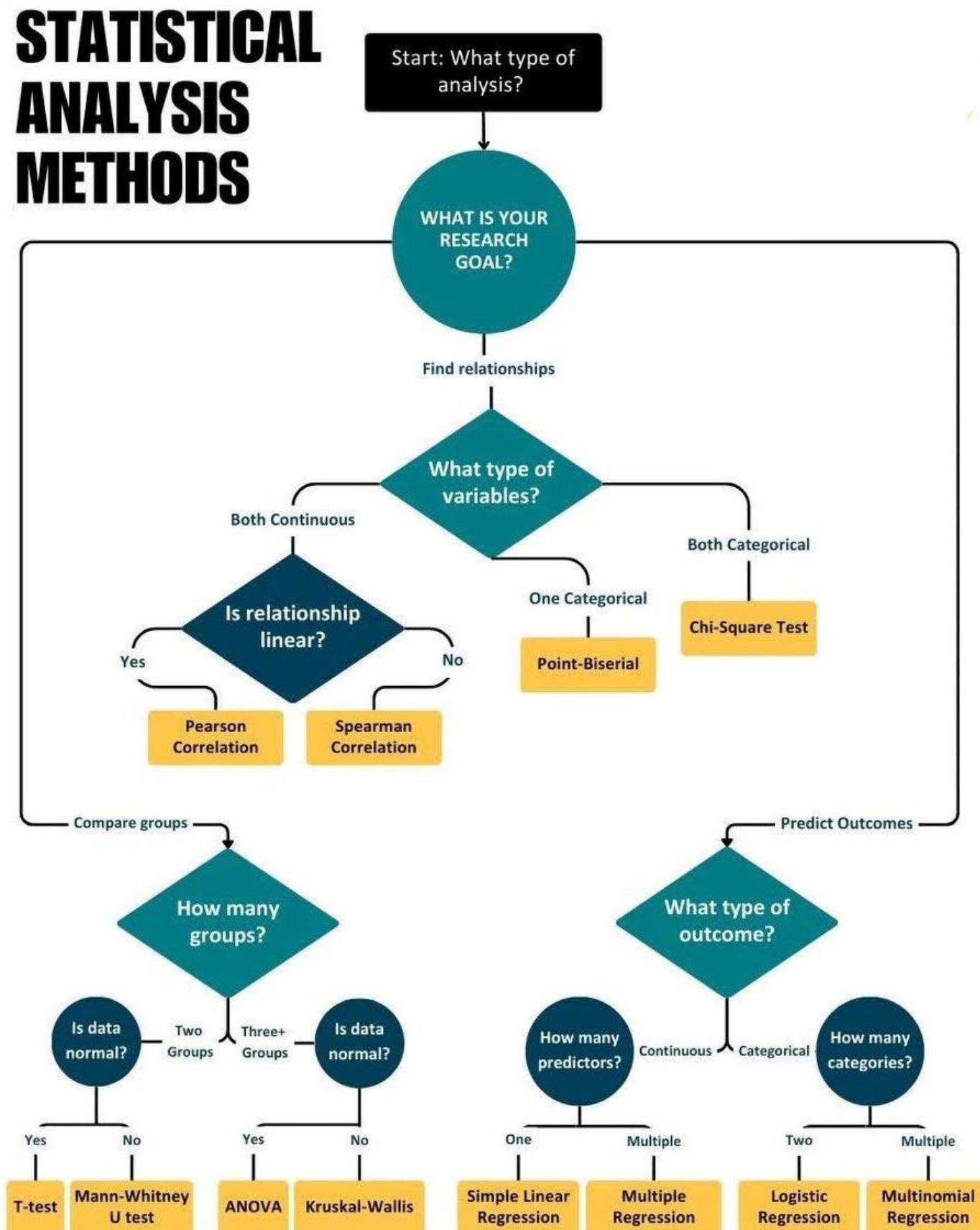
**3. Klastrovanie
(Clustering)**

**4. Redukcia
dimenzií
(Dimensionality
reduction)**

**5. Výber modelu
(Model selection)**

**6.
Predspracovanie
(Preprocessing)**

Výber Štatistickej Metódy



Čo je to Regresia?



- Štatistická technika
- Algoritmus používaný na predpovedanie alebo vizualizáciu a vztah medzi dvoma rôznymi vlastnosťami/premennými
- Na modelovanie vztahu medzi závislou premennou (predikovanou) a 1 alebo viacerými nezávislými premennými (prediktormi)
- Cieľom je predpovedať hodnoty závislej premennej na základe hodnôt nezávislých premenných

Regresia (spojitý výstup)



1. **Predpovedanie cien nehnuteľností:** Určenie ceny domu na základe jeho charakteristík (rozloha, počet izieb, lokalita).
2. **Predikcia predaja:** Odhadovanie predaja výrobkov na základe historických údajov a marketingových kampaní.
3. **Predpovedanie teplôt:** Odhadovanie teploty vzduchu na základe meteorologických údajov a historických trendov.
4. **Analýza výkonnosti:** Predpovedanie výkonu zamestnancov na základe rôznych faktorov, ako sú skúsenosti a vzdelanie.
5. **Finančné analýzy:** Odhadovanie budúcich ziskov spoločnosti na základe minulých výsledkov a trendov v odvetví.
6. **Predikcia výdavkov:** Odhadovanie budúcich výdavkov domácnosti na základe predchádzajúcich nákupných zvyklostí a trendov.

Klasifikácia (diskrétny výstup)



Klasifikácia e-mailov: Rozdelenie e-mailov na spam a ne-spam

Diagnóza chorôb: Určenie, či pacient má určité ochorenie na základe symptómov a testov

Detekcia podvodov: Identifikácia podvodných transakcií na základe vzorcov správania

Rozpoznávanie obrázkov: Klasifikácia obrázkov do kategórií, ako sú zvieratá, rastliny, alebo objekty

Analýza sentimentu: Určenie, či je text pozitívny, negatívny alebo neutrálny

Klasifikácia zákazníkov: Rozdelenie zákazníkov do segmentov na základe ich nákupného správania

Regresia vs. Klasifikácia

Regresia

Predpovedáme spojité, číselné hodnoty (napr. cena, výška, teplota). Výstupom je reálne číslo ($y \in \mathbb{R}$).

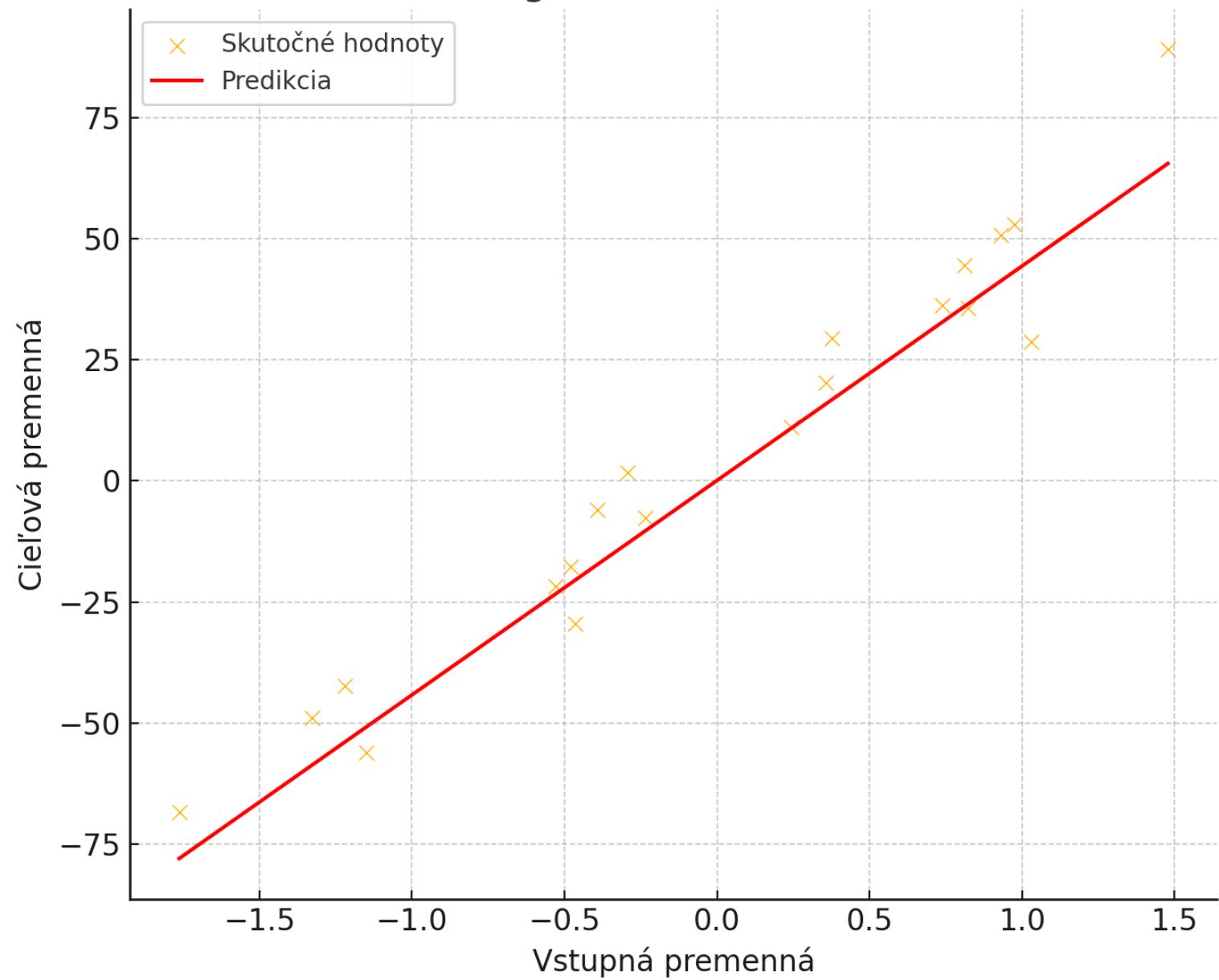
- Predikcia tržieb
- Odhad cien nehnuteľností
- Predpoveď výnosov
- Optimalizácia nákladov

Klasifikácia

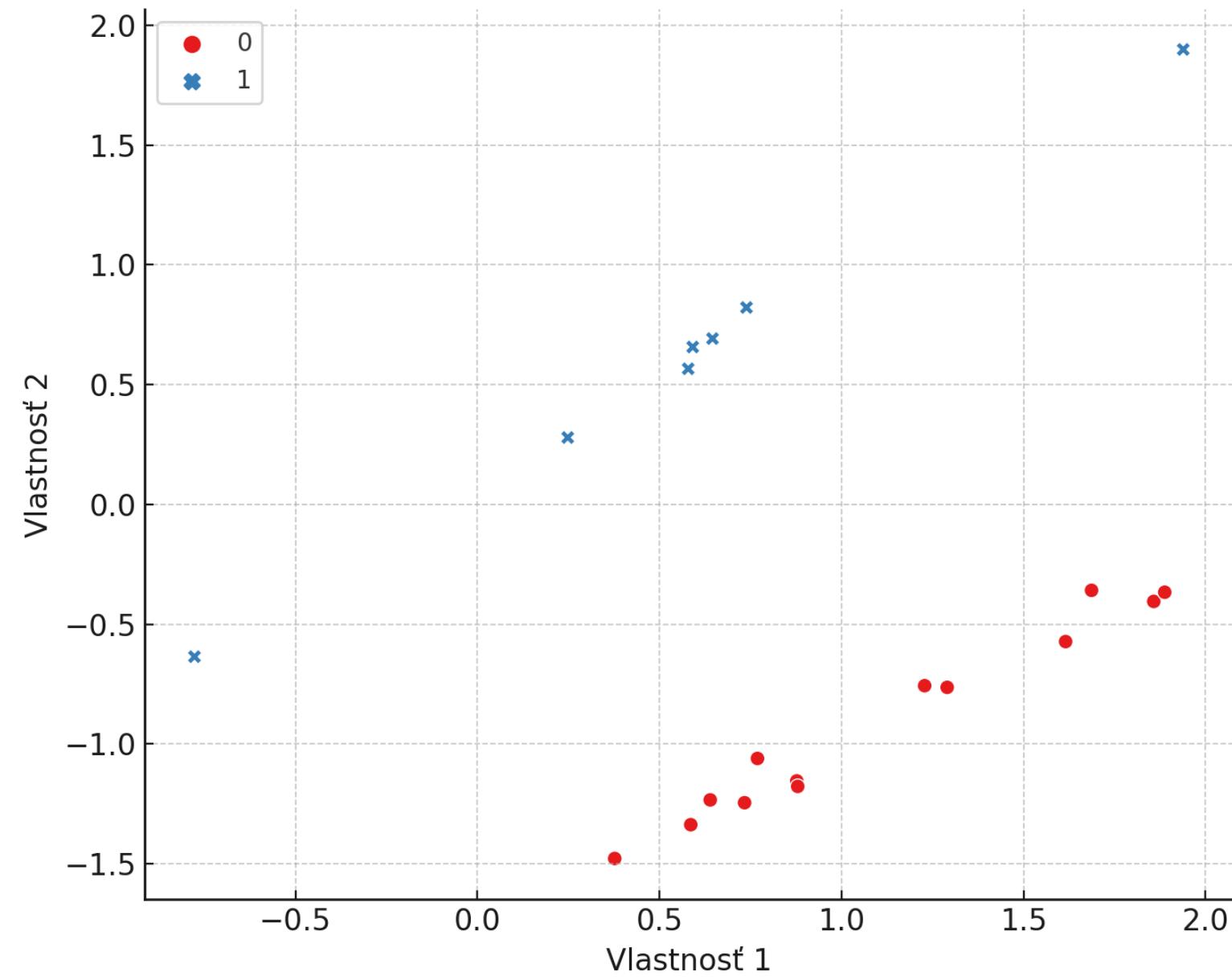
Predpovedáme kategóriu, do ktorej vstup patrí (napr. schválený/neschválený). Výstupom je diskrétna hodnota ($y \in \{0,1,\dots\}$).

- Detekcia spamu
- Diagnostika chorôb
- Rozpoznávanie objektov
- Sentiment analýza

Regresia ($R^2 = 0.94$)



Klasifikácia (Presnosť = 1.00)



Dátové množiny v strojovom učení

Tréningová množina

Množina dát použitá na vytváranie modelu. Model sa na týchto dátach učí rozpoznávať vzory a vzťahy medzi premennými.

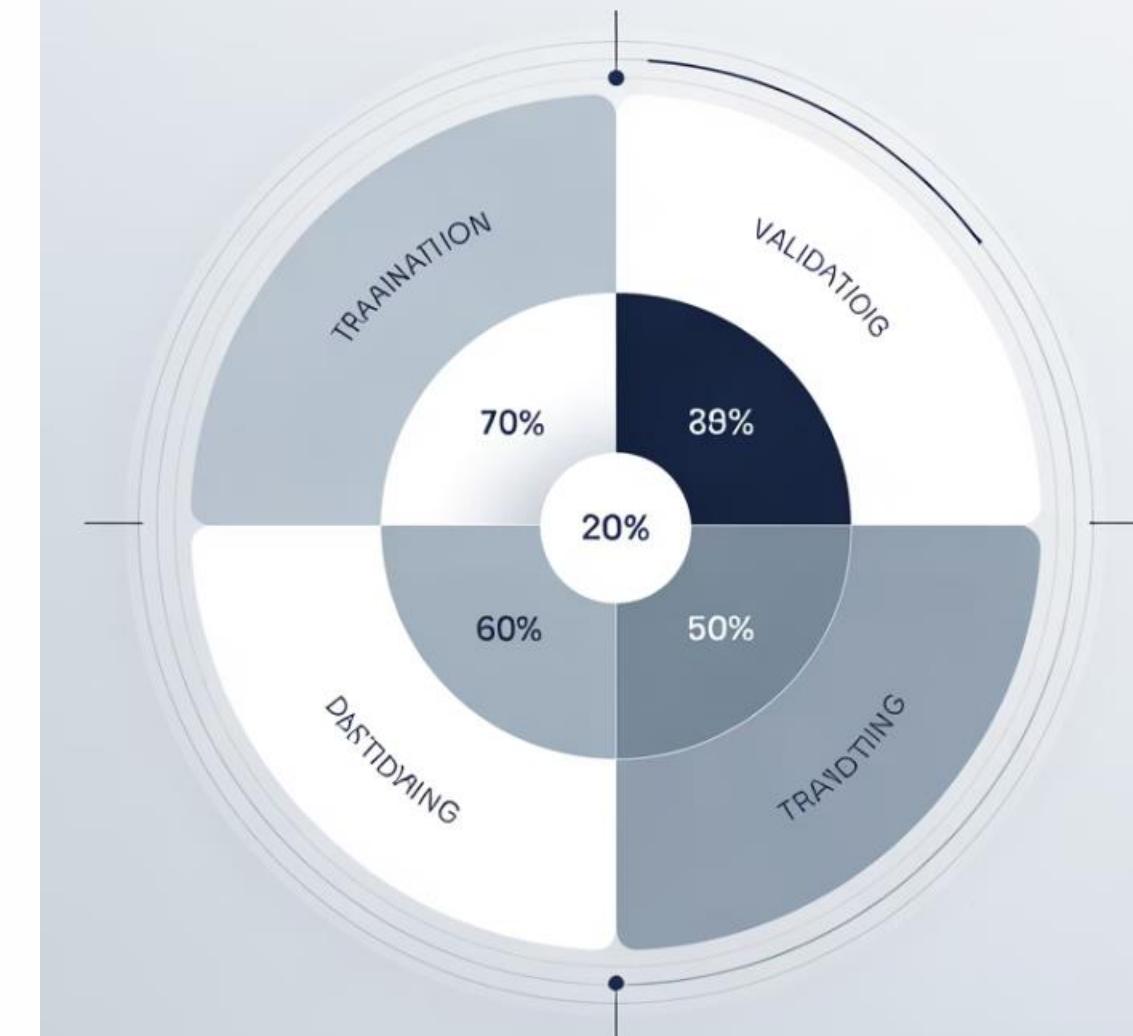
Testovacia množina

Množina dát použitá na vyhodnotenie výkonnosti modelu. Tieto dáta model počas trénovania "nevidel".

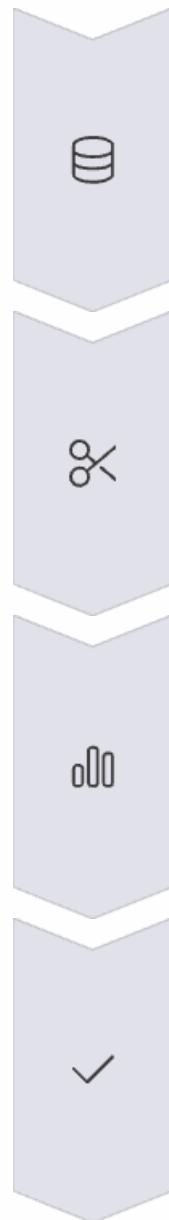
Validačná množina

Voliteľná medzifáza na ladenie parametrov modelu, pomáha predchádzať pretrénovaniu (overfitting).

Understanding
your dataset split



Rozdelenie dát v scikit-learn



Načítanie dát

Získanie dátovej množiny z externého zdroja alebo použitie vstavaných datasetov.

Scikit-learn poskytuje množstvo vstavaných datasetov ako iris, digits, alebo boston. Pre externé dátá môžete použiť pandas: `pd.read_csv()` alebo `pd.read_excel()`.

Rozdelenie dát

Použitie funkcie `train_test_split` na rozdelenie dát na trénovaciu a testovaciu množinu.

Príklad kódu: `X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)`. Parameter `test_size` určuje percentuálnu veľkosť testovacej množiny (20%).

Príprava modelov

Vytvorenie a trénovanie modelov na trénovacej množine.

Modely vytvoríte pomocou tried ako `LinearRegression`, `RandomForestClassifier`, atď.

Trénovanie prebieha metódou `model.fit(X_train, y_train)`, ktorá upraví váhy modelu podľa vzorov v trénovacích dátach.

Vyhodnotenie

Testovanie natrénovaných modelov na testovacej množine.

Vyhodnotenie presnosti modelu pomocou metód ako `model.score(X_test, y_test)`, `mean_squared_error()` alebo `accuracy_score()`. Tieto metriky merajú, ako dobre model generalizuje na nové, nevidené dátá.

Správne rozdelenie dát je klúčové pre vytvorenie robustného modelu. Odporúčaný pomer je typicky 70-80% pre trénovaciu množinu a 20-30% pre testovaciu množinu. Pri väčších datasetoch môžete použiť aj menší pomer testovacej množiny.

Data Science Complete Workflow

[Learn more](#)

```
def load_iris():
    """Load the Iris dataset from sklearn datasets.

    Returns:
        X (array): Features (petal length, petal width).
        y (array): Target variable (iris species).
    """
    from sklearn import datasets
    from sklearn.model_selection import train_test_split

    # Load the Iris dataset
    iris = datasets.load_iris()
    X = iris.data
    y = iris.target

    # Split the data into training and testing sets
    X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

    return X_train, X_test, y_train, y_test
```



Facade tamice the stílo e rgyriečels da tivraistng
endic aue eoder tbses sspbestier cf osoro der a
done wereená vrehirati eo flapbeet treanttienit
avitowide vittneut attstletis.



```
Python
scikit-learn
```

```
Python
scikit-learn
```

Knižnica scikit-learn



Štandard pre ML v Pythone

Rozsiahla knižnica s jednotným API pre rôzne algoritmy strojového učenia. Používa sa v 80% projektov strojového učenia v Pythone. Vyvinutá a udržiavaná komunitou od roku 2007.



Bohatá dokumentácia

Podrobnejšia dokumentácia s príkladmi a tutoriálmi na oficiálnej stránke. Obsahuje aj galériu príkladov s kompletným kódom pre bežné úlohy. Podporuje začiatočníkov aj pokročilých používateľov.



Kompatibilita

Bezproblémová integrácia s numpy, pandas, matplotlib a seaborn. Pracuje s dataframe objektami bez potreby konverzie. Podporuje export modelov pomocou joblib alebo pickle pre použitie v produkčnom prostredí.



Modulárny dizajn

Rozdelenie na moduly pre rôzne úlohy: linear_model, preprocessing, metrics, model_selection, datasets. Umožňuje jednoduché rozšírenie a prispôsobenie pre špecifické potreby. Podporuje vývoj vlastných transformátorov a estimátorov.

Princíp práce so scikit-learn

Vytvorenie modelu

Inicializácia objektu modelu, napr. `model = LinearRegression()`

V scikit-learn každý algoritmus strojového učenia je implementovaný ako Python trieda.

Vytvorením objektu definujeme hyperparametre modelu ešte pred začiatkom trénovania.

Trénovanie modelu

Učenie modelu na trénovacích dátach pomocou metódy `fit()`: `model.fit(X_train, y_train)`

Metóda `fit()` spustí optimalizačný proces, ktorý hľadá najlepšie parametre modelu na základe trénovacích dát. Tento krok je výpočtovo najnáročnejší.

Predikcia

Použitie natrénovaného modelu na predikciu: `y_pred = model.predict(X_test)`

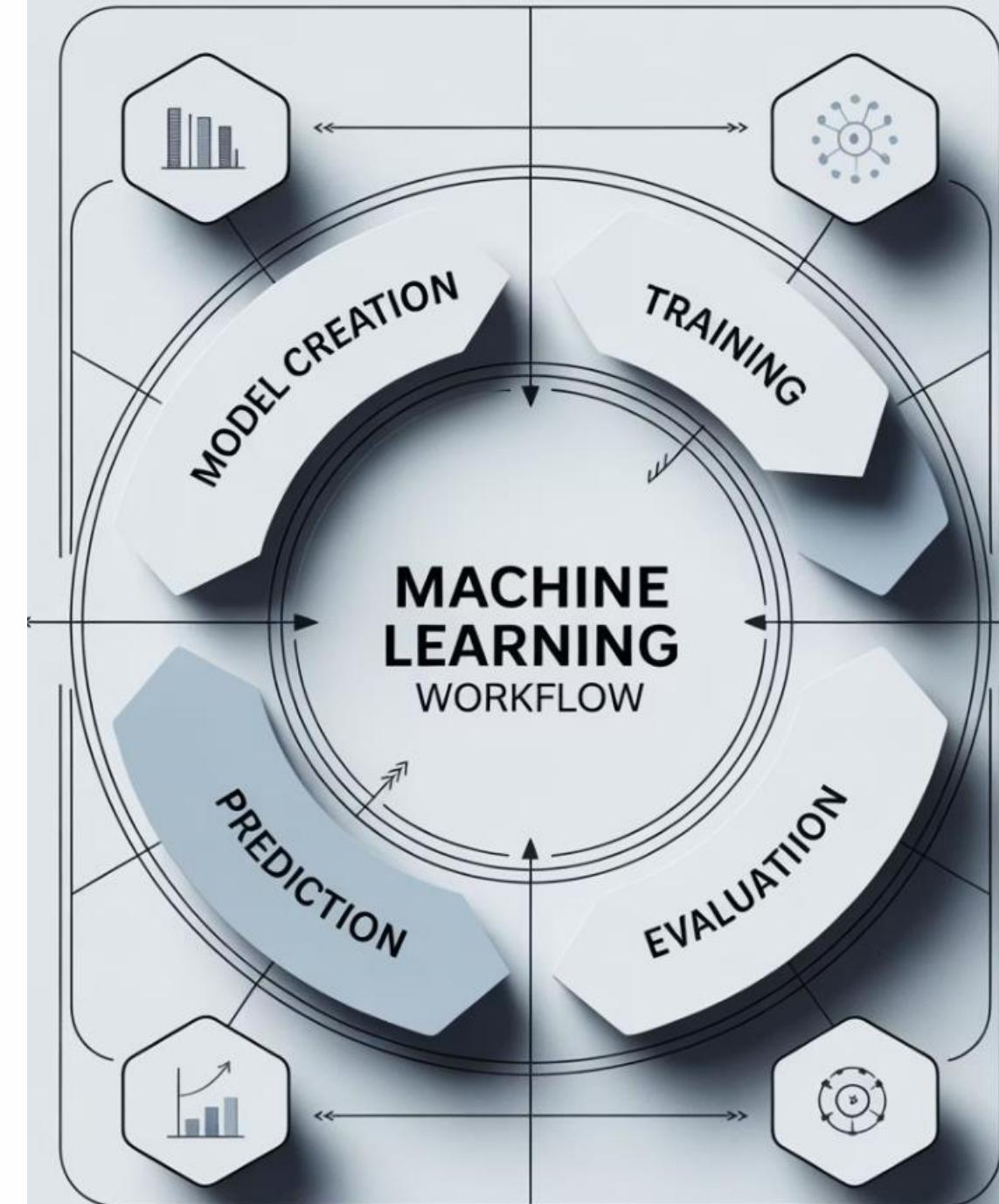
Po natrénovaní model aplikujeme na nové dátá. Model.`predict()` vracia predikcie pre vstupné hodnoty, ktoré model predtým nevidel.

Vyhodnotenie

Meranie výkonu modelu: `score = r2_score(y_test, y_pred)`

Na vyhodnotenie presnosti modelu používame metriky ako R^2 , MSE alebo accuracy.

Modul `metrics` obsahuje funkcie pre rôzne typy hodnotenia podľa typu úlohy (regresia/klasifikácia).





Príprava dát pre modelovanie



Získanie dát

Použitie vstavaných datasetov alebo načítanie externých súborov.

Napríklad: `from sklearn.datasets import load_iris alebo pandas.read_csv("cesta/k/súbor u.csv")` pre načítanie dát.

Čistenie dát

Odstránenie chýbajúcich hodnôt a odľahlých pozorovaní.

Môžeme použiť `df.dropna()` pre odstránenie riadkov s chýbajúcimi hodnotami alebo `df.fillna(hodnota)` pre ich nahradenie. Pre odľahlé pozorovania môžeme použiť IQR alebo Z-skóre.

Škálovanie

Normalizácia a štandardizácia premenných.

Použitie **StandardScaler** pre štandardizáciu (priemer=0, smerodajná odchýlka=1) alebo **MinMaxScaler** pre normalizáciu dát do intervalu [0,1]. Dôležité pre algoritmy citlivé na mierku.

Rozdelenie dát

Separácia na trénovaciu a testovaciu množinu.

S funkciou `train_test_split(X, y, test_size=0.2)` rozdelíme dáta na 80% trénovacích a 20% testovacích. Nastavenie **random_state** zabezpečí reprodukovateľnosť výsledkov.



Získavanie dát v scikit-learn

Predtým ako začneme s modelovaním, musíme získať kvalitné dáta. Scikit-learn ponúka niekoľko možností ako pristupovať k dátam.

Vstavané datasety

Scikit-learn poskytuje niekoľko vstavaných datasetov pre rôzne úlohy strojového učenia:

- `load_diabetes` - dáta o diabete
- `fetch_california_housing` - ceny domov v Kalifornii
- `load_boston` (zastaralý) - ceny domov v Bostonе

Poznámka: Vstavané datasety sú užitočné pre rýchly začiatok a porovnanie algoritmov bez nutnosti hľadať externé zdroje dát.

Externé zdroje

Načítanie dát z externých zdrojov pomocou pandas:

- `pandas.read_csv("data.csv")` - načítanie CSV
- `pandas.read_excel("data.xlsx")` - načítanie Excel súboru
- `pandas.read_sql("SELECT * FROM table", connection)` - načítanie z databázy

Poznámka: Pri práci s externými zdrojmi je dôležité skontrolovať formát dát a ošetriť prípadné chýbajúce hodnoty pred ďalsím spracovaním.

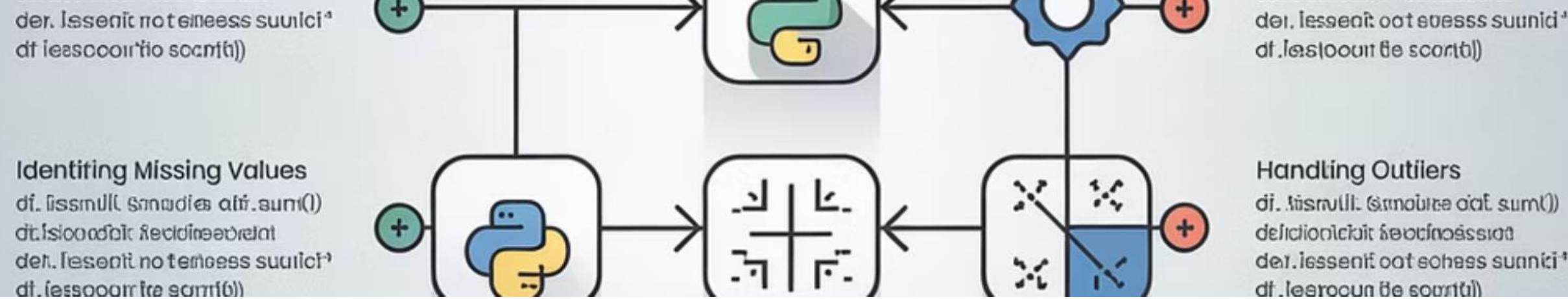
Generovanie dát

Vytvorenie syntetických dát pre testovanie a experimentovanie:

- `make_regression` - generovanie dát pre regresiu
- `make_classification` - generovanie dát pre klasifikáciu
- `make_blobs` - generovanie zhľukov bodov

Poznámka: Syntetické dáta sú výborné pre pochopenie algoritmov a testovanie, ale pre reálne aplikácie je potrebné pracovať so skutočnými dátami.

Pre efektívnu prácu s dátami je dôležité porozumieť ich štruktúre a charakteristikám ešte pred samotnou prípravou pre modelovanie.



Čistenie a spracovanie dát



Identifikácia chýbajúcich hodnôt

Použitie `df.isnull().sum()` na zistenie počtu chýbajúcich hodnôt v každom stĺpci. Tento krok je kľúčový pre pochopenie kvality dát. Ďalšie užitočné funkcie zahŕňajú `df.info()` a `df.describe()`, ktoré poskytujú prehľad o typoch dát a základných štatistikách.



Imputácia chýbajúcich hodnôt

Nahradenie chýbajúcich hodnôt pomocou priemeru, mediánu alebo pokročilejších metód. Scikit-learn poskytuje nástroje ako `SimpleImputer` pre základnú imputáciu a `IterativeImputer` pre pokročilejšie metódy využívajúce predikciu. Pri kategoriálnych premenných môžeme použiť najčastejšiu hodnotu alebo špeciálnu kategóriu označujúcu chýbajúce údaje.



Odstránenie chýbajúcich hodnôt

Použitie `df.dropna()` na odstránenie riadkov s chýbajúcimi hodnotami. Môžeme špecifikovať subset stĺpcov pomocou parametra `subset` alebo odstrániť len stĺpce s chýbajúcimi hodnotami pomocou `axis=1`. Tento prístup je vhodný, keď je počet chýbajúcich hodnôt malý v porovnaní s celkovým objemom dát.



Odstránenie odľahlých pozorovaní

Identifikácia a odstránenie extrémnych hodnôt, ktoré môžu skresliť model. Na detekciu odľahlých hodnôt môžeme použiť metódy ako IQR (medzikvartilové rozpätie), Z-skóre alebo algoritmy ako Isolation Forest a DBSCAN. Pred odstránením je dobré vizualizovať dáta pomocou krabicových grafov alebo histogramov pre lepšie pochopenie distribúcie.

Škálovanie a normalizácia dát

Prečo škálovať dát?

Mnohé algoritmy strojového učenia sú citlivé na rozsah vstupných premenných. Premenné s väčším rozsahom môžu dominovať pri výpočte vzdialenosí alebo gradientov.

Škálovanie zabezpečuje, že všetky premenné majú podobný vplyv na model, čo viedie k lepšej konvergencii a stabilite.

Metódy škálovania v scikit-learn

- **StandardScaler** - štandardizácia (z-skóre): $(x - \mu) / \sigma$
- **MinMaxScaler** - normalizácia do rozsahu [0, 1]
- **RobustScaler** - robustné škálovanie odolné voči odľahlým hodnotám
- **Normalizer** - normalizácia riadkov na jednotkovú normu

Rozdelenie dát na trénovaciu a testovaciu množinu

1

Kompletný dataset

Všetky dostupné dáta pred rozdelením



Rozdelenie pomocou train_test_split

Náhodné rozdelenie dát na trénovaciu a testovaciu množinu



Trénovanie na trénovacej množine

Model sa učí len na trénovacích dátach



Testovanie na testovacej množine

Vyhodnotenie výkonu na nevidených dátach

Poznámky k rozdeleniu dát:

- **Štandardné rozdelenie:** Typicky sa používa pomer 80% trénovacích a 20% testovacích dát, prípadne 70/30 pre menšie datasety
- **Stratifikované rozdelenie:** Pre klasifikačné úlohy je dôležité zachovať rovnaké zastúpenie tried v oboch množinách pomocou parametra `stratify`
- **Validačná množina:** Okrem trénovacej a testovacej množiny je často potrebná aj validačná množina na ladenie hyperparametrov
- **Cross-validácia:** Pre robustnejšie vyhodnotenie modelov sa odporúča použiť K-fold cross-validáciu
- **Časové rady:** Pri časových radách nie je vhodné náhodné rozdelenie, ale chronologické (staré dáta na trénovanie, nové na testovanie)



Čo sa naučíme?

1. Čo je strojové učenie (machine learning) a prečo sa používa?
2. Aké poznáme typy učenia?
3. Čo znamená supervised learning?
4. Čo je regresia v strojovom učení?
5. Aký je rozdiel medzi regresiou a klasifikáciou?
6. Na čo slúži knižnica scikit-learn?
7. Profit

Aké máme Typy a Modely Regresíí?

AKREDITOVANÝ KURZ





Čo sa naučíme?

1. Aké máme typy regresií a ako sa používajú v praxi?
2. Kedy sa používa polynomiálna regresia?
3. Čo je Ridge a Lasso regresia?
4. Aký je rozdiel medzi lineárhou a logistickou regresiou?
5. Ako si pripraviť dáta na regresiu?
6. Aký typ regresie je vhodný pre reálne problémy?
7. Profit

Typy regresných modelov

Lineárna regresia

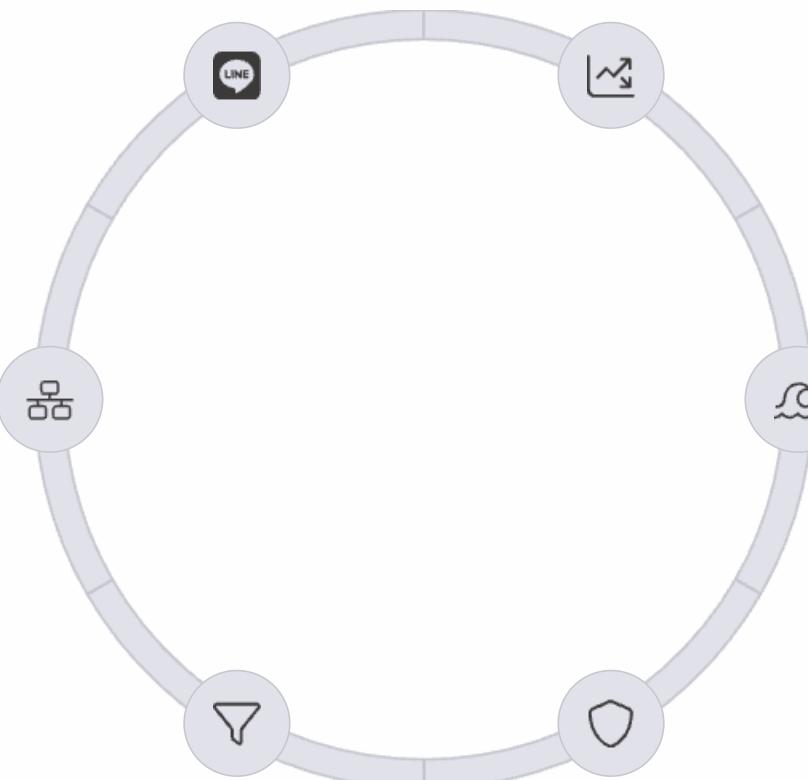
Jednoduchý model predpokladajúci lineárny vztah medzi premennými. Matematicky vyjadrený ako $y = \beta_0 + \beta_1 x + \varepsilon$, kde β_0 je intercept, β_1 je smernica a ε je chyba.

ElasticNet

Kombinácia L1 a L2 regularizácie. Spája výhody Ridge a Lasso regresie. Penalizačný člen má tvar $\lambda_1 \sum |\beta| + \lambda_2 \sum \beta^2$. Mimoriadne užitočná pri práci s korelovanými premennými.

Lasso regresia

L1 regularizácia, pomáha s výberom dôležitých premenných. Pridáva penalizačný člen $\lambda \sum |\beta|$ do funkcie straty. Automaticky vykonáva výber premenných tým, že niektoré koeficienty nastaví presne na 0.



Viacnásobná regresia

Rozšírenie lineárnej regresie s viacerými vstupnými premennými. Model má tvar $y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \dots + \beta_n x_n + \varepsilon$. Umožňuje analyzovať komplexnejšie vztahy.

Polynomiálna regresia

Nelineárny vztah, rozšírenie lineárnej regresie. Používa sa, keď dáta vykazujú zakrivený vztah. Model má tvar $y = \beta_0 + \beta_1 x + \beta_2 x^2 + \dots + \beta_n x^n + \varepsilon$.

Ridge regresia

L2 regularizácia, znižuje riziko pretrénovania. Pridáva penalizačný člen $\lambda \sum \beta^2$ do funkcie straty. Vhodná pre multikolinearitu a pri veľkom počte prediktorov.

Python Strojové Učenie Algoritmy a Zložitosť

Time Complexity

n: data size l: Number of iterations
p: Number of features m: Number of components
T: Number of trees h: Number of hidden units
k: number of clusters

ML Algorithms	Training Time	Inference Time
Linear Regression	$O(np^2 + p^3)$	$O(p)$
Logistic Regression	$O(np^2 + p^3)$	$O(p)$
Naive Bayes	$O(np)$	$O(p)$
Decision Tree	Avg: $O(T \cdot n \log n)$ Worst: $O(n^2)$	Avg: $O(T \cdot n \log n)$ Worst: $O(n)$
Random Forest	$O(T \cdot n \log n)$	$O(T \cdot \log n)$
Gradient Boosted Trees	$O(T \cdot n \log n)$	$O(T \cdot \log n)$
Principal Component	$O(np^2 + p^3)$	$O(pm)$
K-Nearest Neighbor	$O(1)$	$O(np)$
K-Means	$O(l \cdot k \cdot n \cdot p)$	$O(k \cdot p)$
Dense Neural Networks	$O(l \cdot n \cdot p \cdot h)$	$O(p \cdot h)$

Základné Typy Regresie

1. Lineárna regresia

- Modeluje vzťah medzi závislou a nezávislou premennou ako priamku
- $Y = \beta_0 + \beta_1 X + \epsilon$
- Predpovedanie ceny domu (Y) na základe jeho veľkosti (X)
- Používa sa, keď existuje očakávaný lineárny vzťah medzi 2 premennými

2. Viacnásobná lineárna regresia

- Modeluje vzťah medzi závislou premennou a viacerými nezávislými premennými
- $Y = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \dots + \beta_n X_n + \epsilon$
- Predpovedanie ceny domu (Y) na základe viacerých faktorov, ako sú jeho veľkosť (X_1), lokalita (X_2), počet izieb (X_3), a vek domu (X_4)
- Používa, keď je viacero faktorov (nezávislých premenných), ktoré ovplyvňujú predikciu závislej premennej

3. Logistická regresia

- Používa sa, keď je závislá premenná kategóriová (napríklad "áno" alebo "nie")
- Predpovedanie pravdepodobnosti, že zákazník vykoná nákup ($Y = 1$) alebo nie ($Y = 0$) na základe jeho demografických údajov (X)

Pokroč. Typy Regresie

4. Polynomiálna regresia

- Používa sa na modelovanie nelineárnych vzťahov medzi nezávislými a závislými premennými, keď lineárny model nedáva uspokojivé výsledky.
- $Y = \beta_0 + \beta_1 X + \beta_2 X^2 + \beta_3 X^3 + \dots + \beta_n X^n + \epsilon$
- Predpovedanie cenovej závislosti akcie na čase, kde vzťah medzi časom a cenou nie je lineárny.
- Tento typ regresie sa používa, keď vzťah medzi premennými nie je lineárny, ale môže byť prispôsobený nelineárnym polynomiálnym vzorcом

5. Ridge regresia

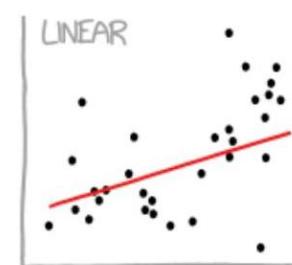
- Je variantom viacnásobnej lineárnej regresie, ktorý pridáva regulárizáčny člen, aby sa zabránilo overfittingu (prílišné prispôsobenie modelu dátam).
- Pri modelovaní s viacerými nezávislými premennými, kde je riziko, že model bude príliš presný (overfitting).
- Používa sa pri veľkých datasetoch, kde sa obávame multikolinearít alebo overfittingu

6. Lasso regresia

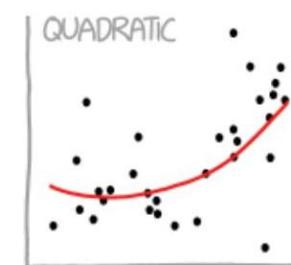
- Podobná ridge regresii, ale namiesto štvorcov koeficientov používa L1 reguláciu, čo spôsobuje, že niektoré koeficienty regresie môžu byť presne nula. To umožňuje selekciu vlastností.
- Lasso je vhodné, keď máme mnoho premenných a chceme sa zamerať len na najvýznamnejšie.

CURVE-FITTING METHODS AND THE MESSAGES THEY SEND

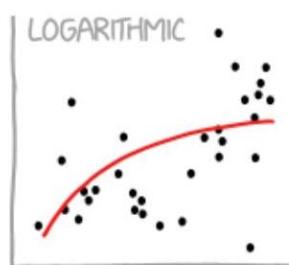
Typy Regresii



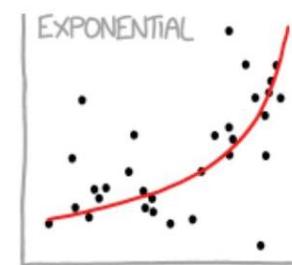
"HEY, I DID A REGRESSION."



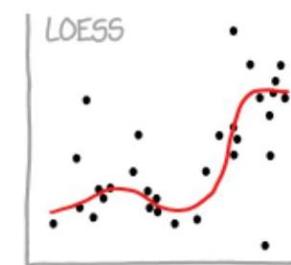
"I WANTED A CURVED LINE, SO I MADE ONE WITH MATH."



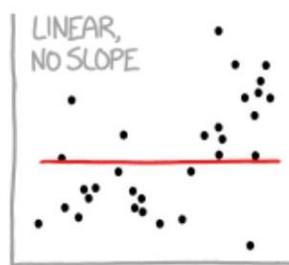
"LOOK, IT'S TAPERING OFF!"



"LOOK, IT'S GROWING UNCONTROLLABLY!"



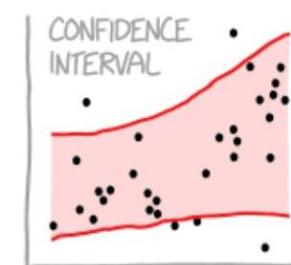
"I'M SOPHISTICATED, NOT LIKE THOSE BUMBLING POLYNOMIAL PEOPLE."



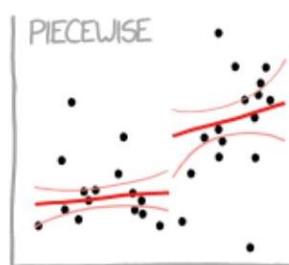
"I'M MAKING A SCATTER PLOT BUT I DON'T WANT TO."



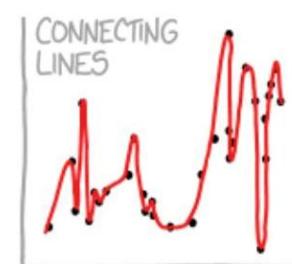
"I NEED TO CONNECT THESE TWO LINES, BUT MY FIRST IDEA DIDN'T HAVE ENOUGH MATH."



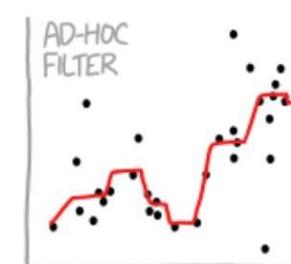
"LISTEN, SCIENCE IS HARD. BUT I'M A SERIOUS PERSON DOING MY BEST."



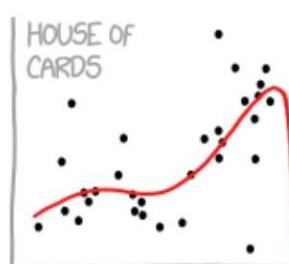
"I HAVE A THEORY, AND THIS IS THE ONLY DATA I COULD FIND."



"I CLICKED 'SMOOTH LINES' IN EXCEL."



"I HAD AN IDEA FOR HOW TO CLEAN UP THE DATA. WHAT DO YOU THINK?"



"AS YOU CAN SEE, THIS MODEL SMOOTHLY FITS THE- WAIT NO NO DON'T EXTEND IT AAAAAAA!!"



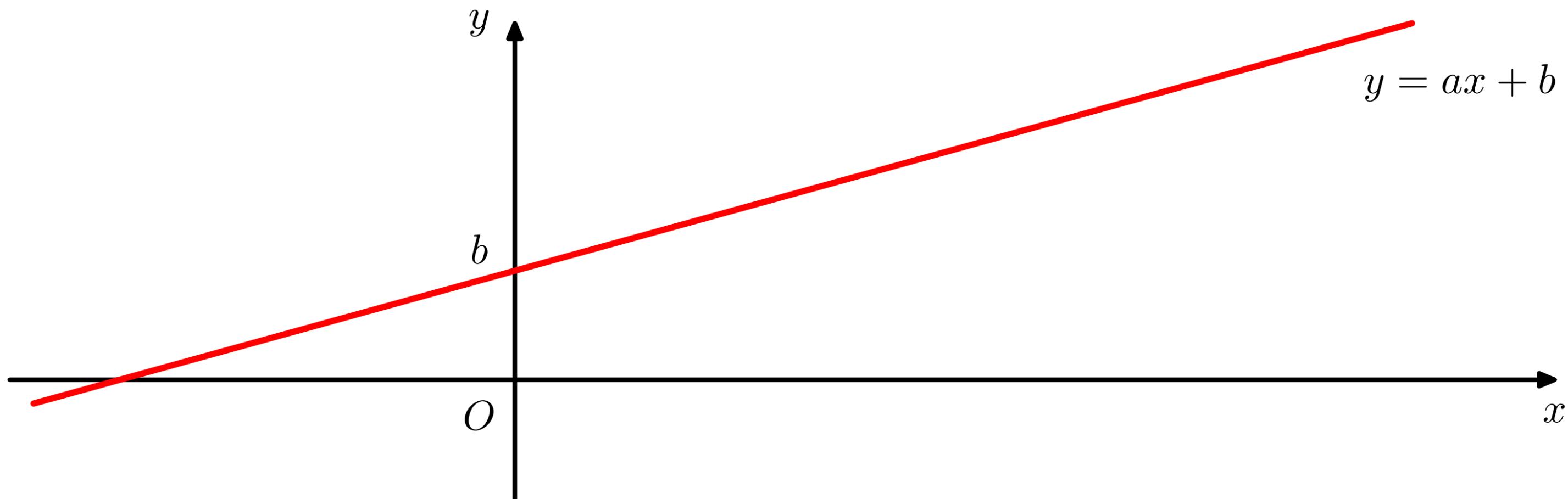
fetch_california_housing

Stĺpec	Význam	Typ	Príklad hodnoty
MedInc	Medián príjmu v štvrti (v 10 000 \$)	číselný	3.87 = 38 700 \$
HouseAge	Priemerný vek domov v štvrti (v rokoch)	číselný	21
AveRooms	Priemerný počet izieb na domácnosť	číselný	5.4
AveBedrms	Priemerný počet spální na domácnosť	číselný	1.1
Population	Počet obyvateľov v štvrti	číselný	850
AveOccup	Priemerný počet osôb na domácnosť	číselný	2.8
Latitude	Zemepisná šírka (lokácia v Kalifornii)	číselný	34.25
Longitude	Zemepisná dĺžka (lokácia v Kalifornii)	číselný	-118.5
MedHouseVal	🎯 Cieľová premenná: mediánová hodnota domu (v 100 000 \$)	číselný	1.78 = 178 000 \$

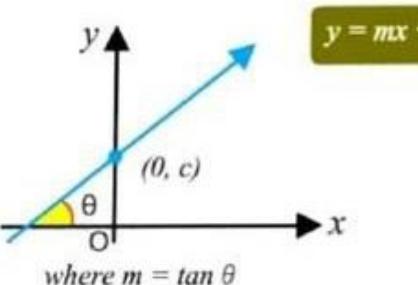
Lineárna funkcia –
$$f : y = ax + b, \quad a, b \in \mathbb{R}, \quad a \neq 0$$

$$\mathcal{D}(f) = \mathbb{R}, \quad \mathcal{H}(f) = \mathbb{R}.$$

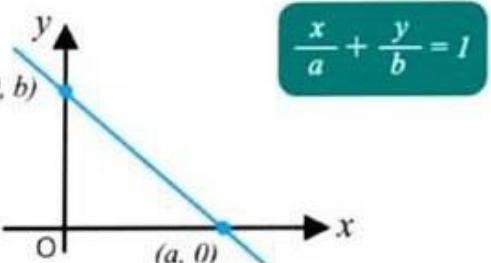
Grafom je priamka so smernicou a , ktorá na osi y vytína úsek b .



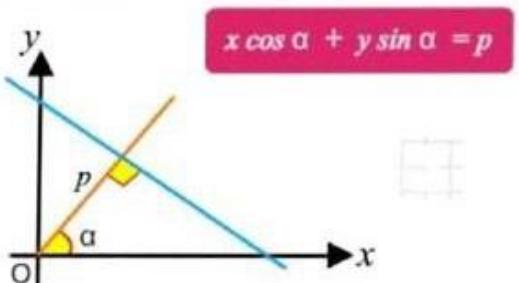
1 Slope - Intercept Form



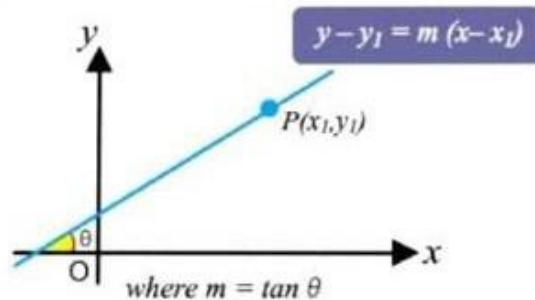
2 Double Intercept Form



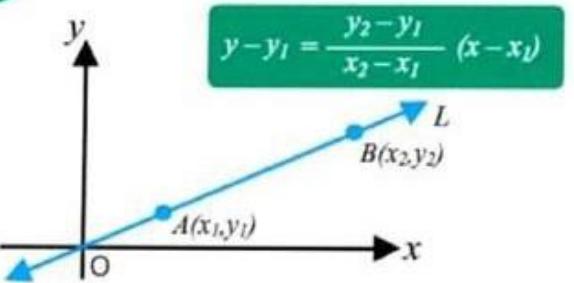
3 Normal Form



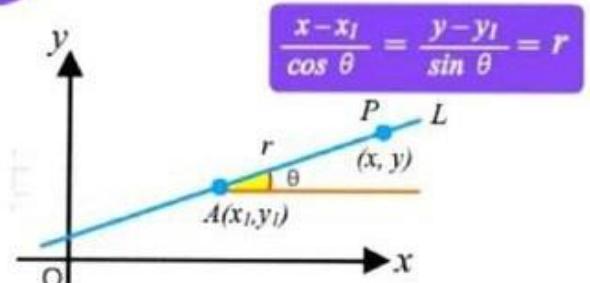
4 Slope - Point Form



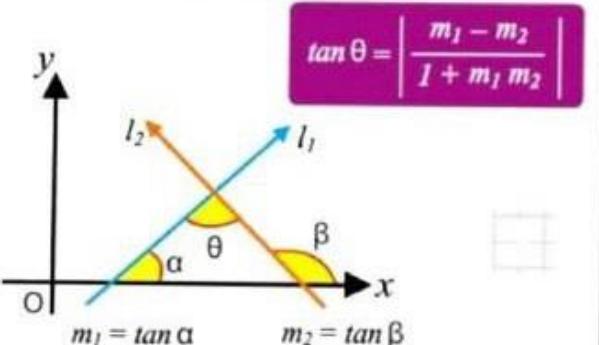
5 Two Point Form



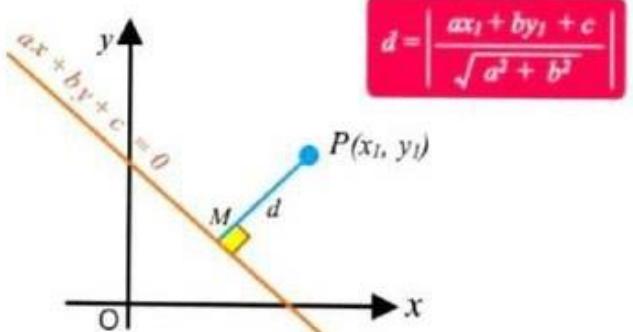
6 Parametric Form



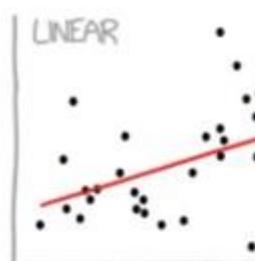
7 Angle between two Straight lines



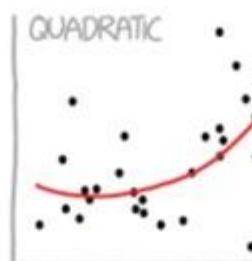
8 Distance between Point & line



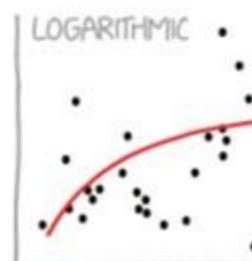
CURVE-FITTING METHODS AND THE MESSAGES THEY SEND



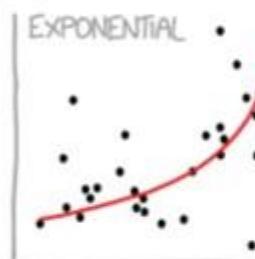
"HEY, I DID A REGRESSION."



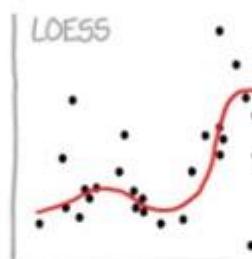
"I WANTED A CURVED LINE, SO I MADE ONE WITH MATH."



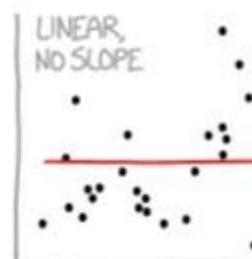
"LOOK, IT'S TAPERING OFF!"



"LOOK, IT'S GROWING UNCONTROLLABLY!"



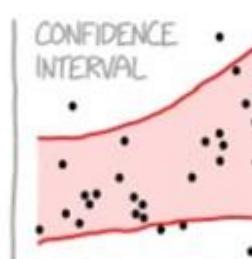
"I'M SOPHISTICATED, NOT LIKE THOSE BUMBLING POLYNOMIAL PEOPLE."



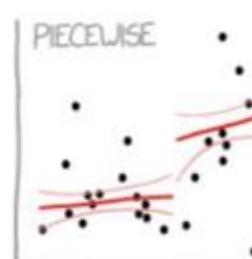
"I'M MAKING A SCATTER PLOT BUT I DON'T WANT TO."



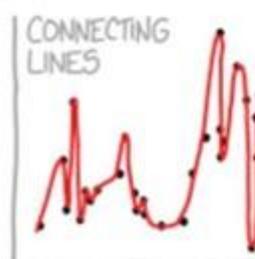
"I NEED TO CONNECT THESE TWO LINES, BUT MY FIRST IDEA DIDN'T HAVE ENOUGH MATH."



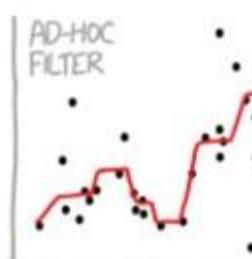
"LISTEN, SCIENCE IS HARD. BUT I'M A SERIOUS PERSON DOING MY BEST."



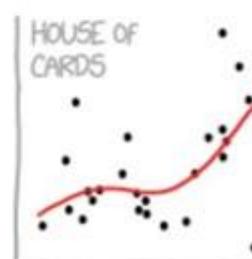
"I HAVE A THEORY, AND THIS IS THE ONLY DATA I COULD FIND."



"I CLICKED 'SMOOTH LINES' IN EXCEL."

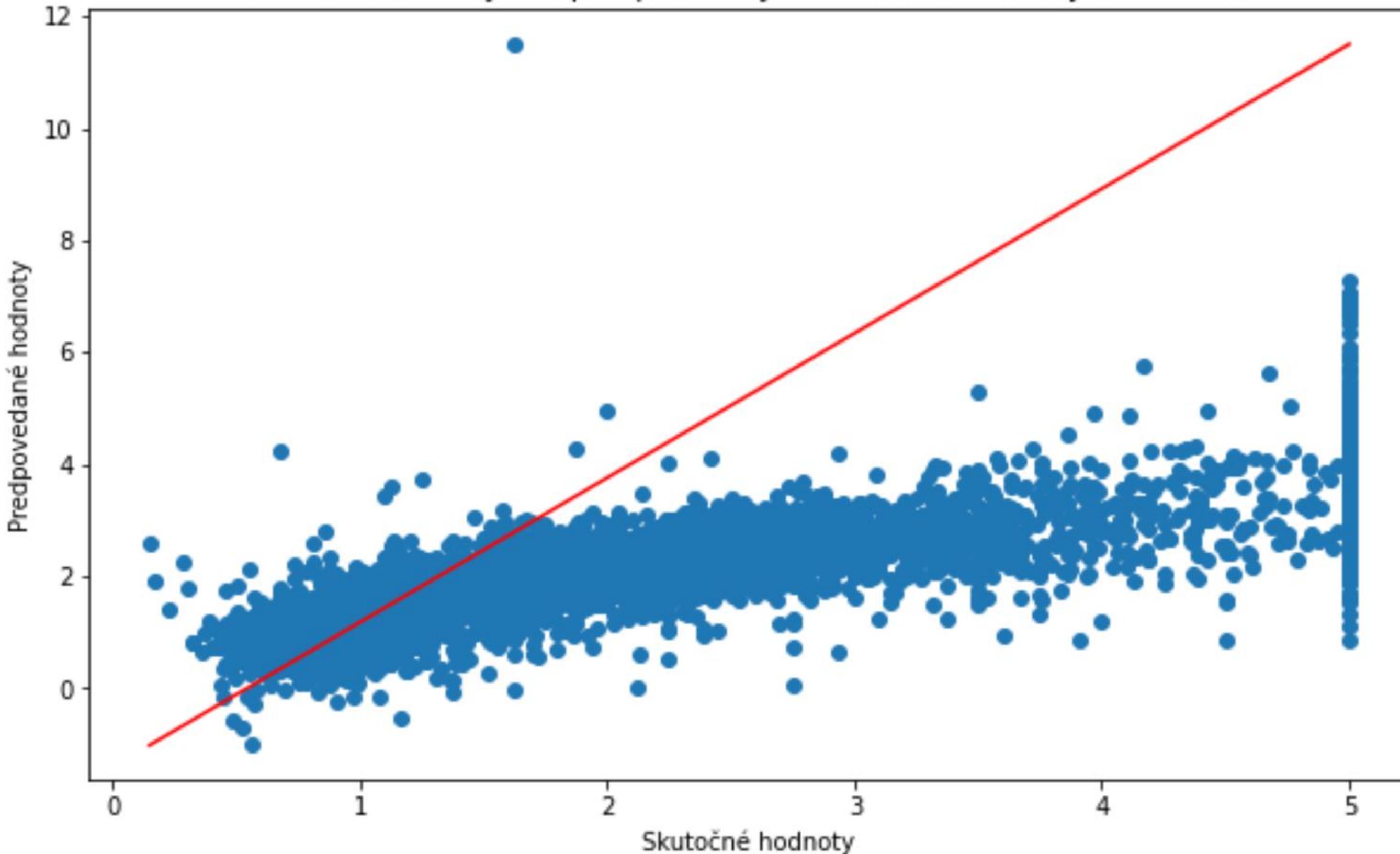


"I HAD AN IDEA FOR HOW TO CLEAN UP THE DATA. WHAT DO YOU THINK?"

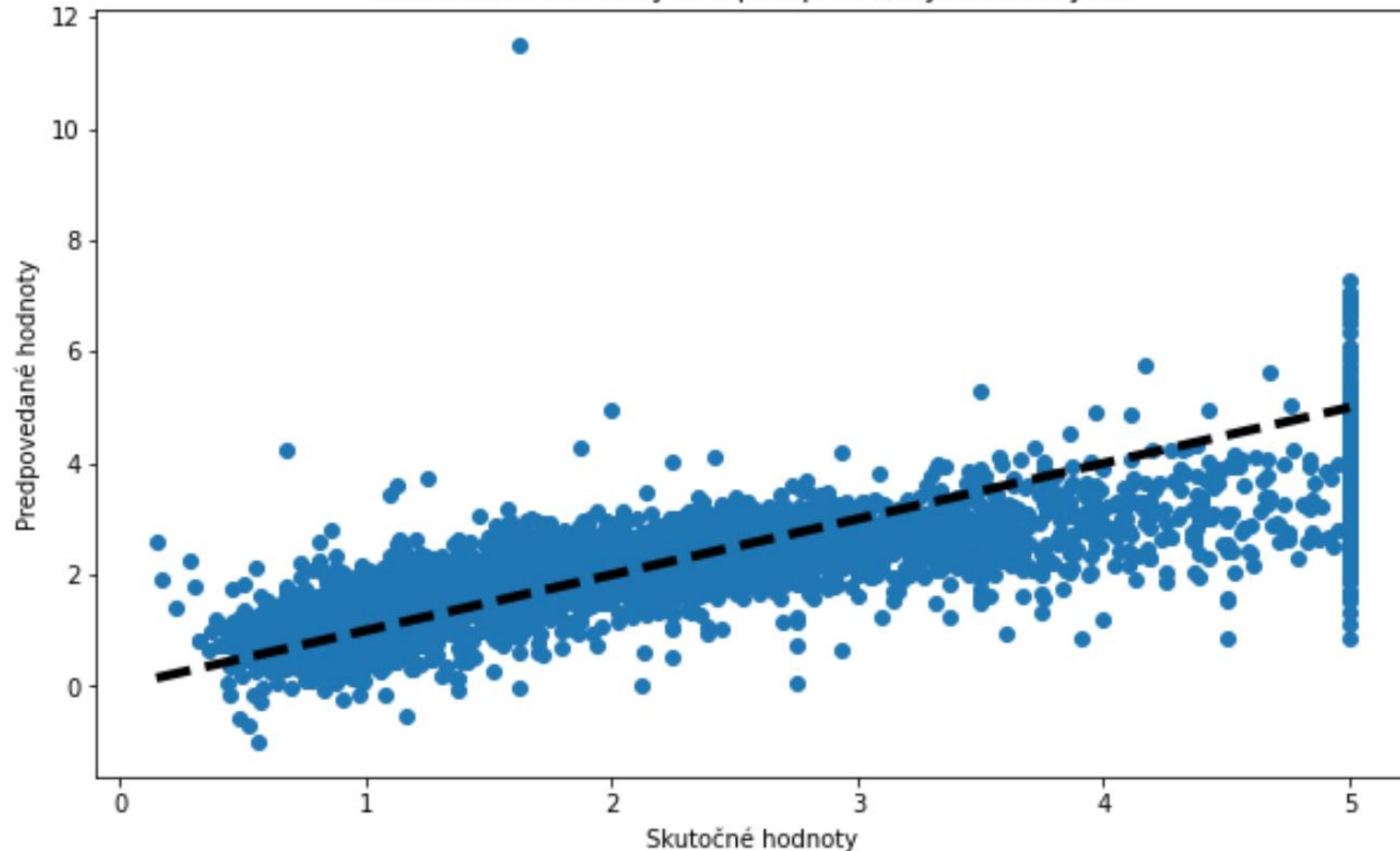


"AS YOU CAN SEE, THIS MODEL SMOOTHLY FITS THE- WAIT NO NO DON'T EXTEND IT AAAAAAA!!"

Porovnanie skutočných a predpovedaných hodnôt mediánových cien domov



Porovnanie skutočných a predpovedaných cien bytov



Vyhodnocovanie regresného modelu

R² (koeficient determinácie)

Miera, ktorá udáva podiel vysvetlenej variability na celkovej variabilite. Hodnoty sa pohybujú od 0 do 1, pričom vyššie hodnoty znamenajú lepší model.

```
r2 = r2_score(y_test, y_pred)
```

Poznámka: Hodnoty nad 0.7 sa považujú za dobré. R² môže byť aj záporný pri veľmi zlých modeloch. Vhodný pre porovnanie rôznych modelov na rovnakých dátach.

MSE (priemerná štvorcová chyba)

Priemerná hodnota štvorcov rozdielov medzi predikovanými a skutočnými hodnotami. Nižšie hodnoty znamenajú lepší model.

```
mse = mean_squared_error(y_test, y_pred)
```

Poznámka: Dáva väčšiu váhu veľkým chybám. Jednotka metriky je kvadrát jednotky predikovanej premennej. Užitočné pri optimalizácii, ale ťažšie interpretovateľné.

MAE (priemerná absolútна chyba)

Priemerná hodnota absolútnych rozdielov medzi predikovanými a skutočnými hodnotami. Nižšie hodnoty znamenajú lepší model.

```
mae = mean_absolute_error(y_test, y_pred)
```

Poznámka: Ľahšie interpretovateľná metrika než MSE, keďže používa rovnakú jednotku ako predikovaná premenná. Robustnejšia voči odľahlým hodnotám. Vhodná pri potrebe zrozumiteľnej interpretácie chyby.

Regression MODEL EVALUATION METRICS



Regresný Koeficient Determin.

R^2



- **Štatistická metrika**
- Vyjadruje, **ako dobre regresný model vysvetľuje rozptyl závislej premennej**
- **Do akej miery môžeme dôverovať modelu pri predpovedaní?**
- **Vlastnosti:**
 - Rozsah hodnôt: $0 \leq R^2 \leq 1$
 - $R^2 = 1$ model dokonale vysvetľuje všetky odchýlky (ideálny prípad)
 - $R^2 = 0$ model nič nevysvetľuje, predikcie sú ako náhodné
 - $R^2 \approx 0,7$ 70 % variability závislej premennej vysvetlených modelom
 - $R^2 < 0,3$ slabá vysvetľovacia schopnosť modelu
 - Ak $R^2 = 0,85 \rightarrow 85\%$ rozdielov v predajoch vysvetľuje reklama a len 15 % je spôsobených inými faktormi (napr. sezónnosťou, konkurenciou).

Regresný Koeficient Determin.

R^2



- **Vlastnosti:**
 - Hodnota sa pohybuje v rozmedzí 0 až 1
 - $R^2 \approx 1 \rightarrow$ model veľmi dobre vysvetľuje variabilitu závislej premennej
 - $R^2 \approx 0 \rightarrow$ model nevysvetľuje žiadnu variabilitu, predikcia je náhodná
 - Vyjadruje podiel rozptylu vysvetleného modelom
 - Citlivý na počet nezávislých premenných a pretrénovanie modelu (overfitting)
 - Platí pre lineárnu aj viacnásobnú regresiu
- **Použitie:**
 - Hodnotenie presnosti predikčných modelov
 - Vyhodnocovanie lineárnych regresných vzťahov v štatistike
 - Odhad účinnosti vysvetľujúcich premenných pri predikcii výsledkov (napr. predaj vs. cena)
- **Príklad:**
 - Ak model predpovedá predaj produktu na základe vyšky investície do reklamy a $R^2 = 0,85$, znamená to, že 85 % variability predaja možno vysvetliť výškou reklamy.
 - Zvyšných 15 % ovplyvňujú iné faktory (napr. konkurencia, sezóna).

Vlastnosť	r (korelácia)	R ² (determinácia)
Rozsah hodnôt	-1 až +1	0 až 1
Smer vztahu	Áno (kladný alebo záporný)	Nie (vždy nezáporné)
Typ analýzy	Vztah	Presnosť vysvetlenia/predikcie
Lineárnosť	Meria priamo	Odvodzuje sa z r ²
Použitie	Korelačná analýza	Regresná analýza

Regresný Koeficient Determin. R^2

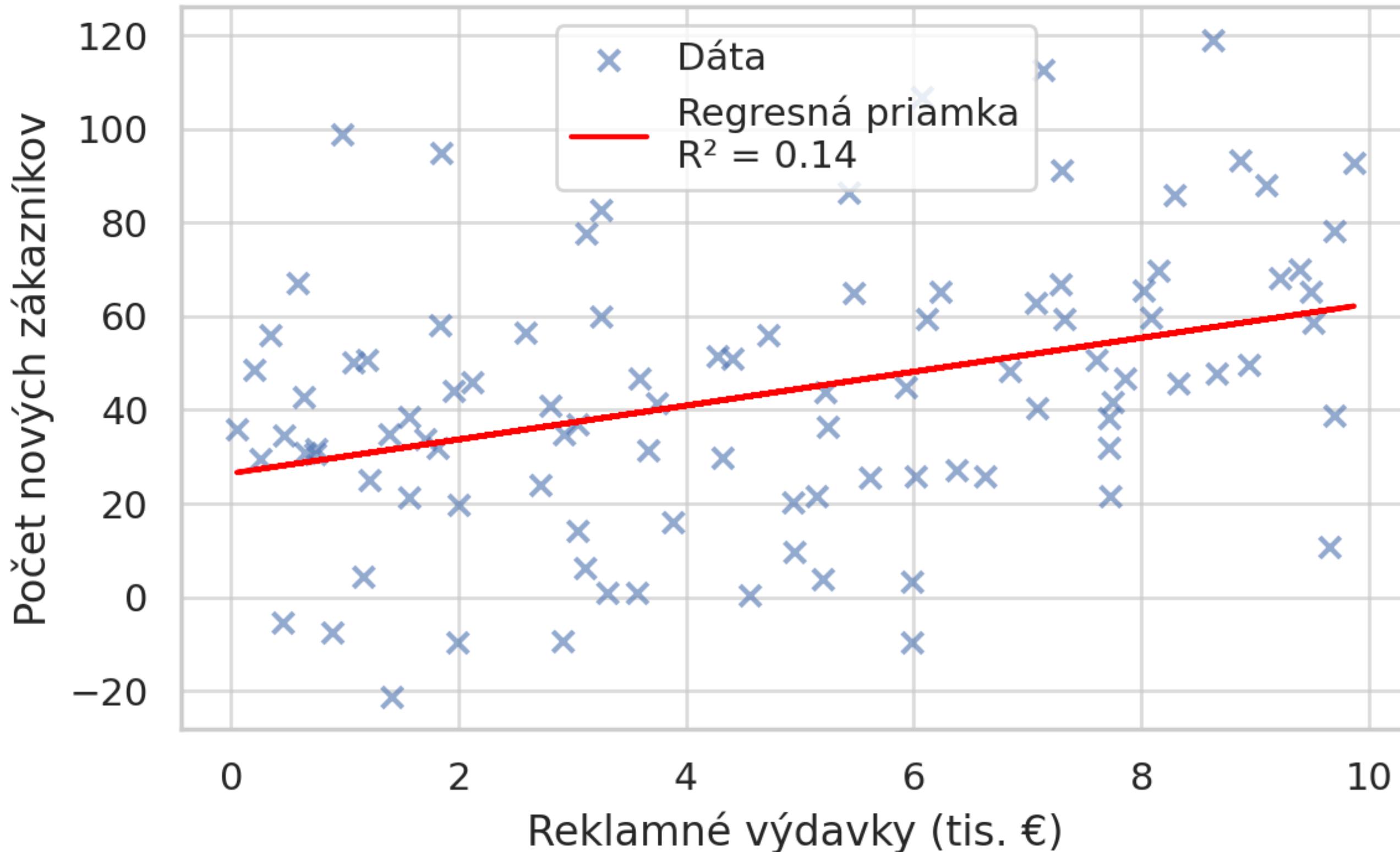
-  **Veľkosť databázy a čas zálohovania**
 - Skúmanie vzťahu medzi veľkosťou databázy (v GB) a časom potrebným na zálohovanie (v min.)
 - Ak $R^2 \approx 0,85$ → väčšie databázy majú lineárne vyšší čas zálohovania

-  **Počet commitov a výskyt bugov**
 - Analýza, či viac commitov (zmien v kóde) vede k vyššiemu počtu bugov
 - Ak $R^2 \approx 0,4$ → slabšia pozitívna lineárna závislosť

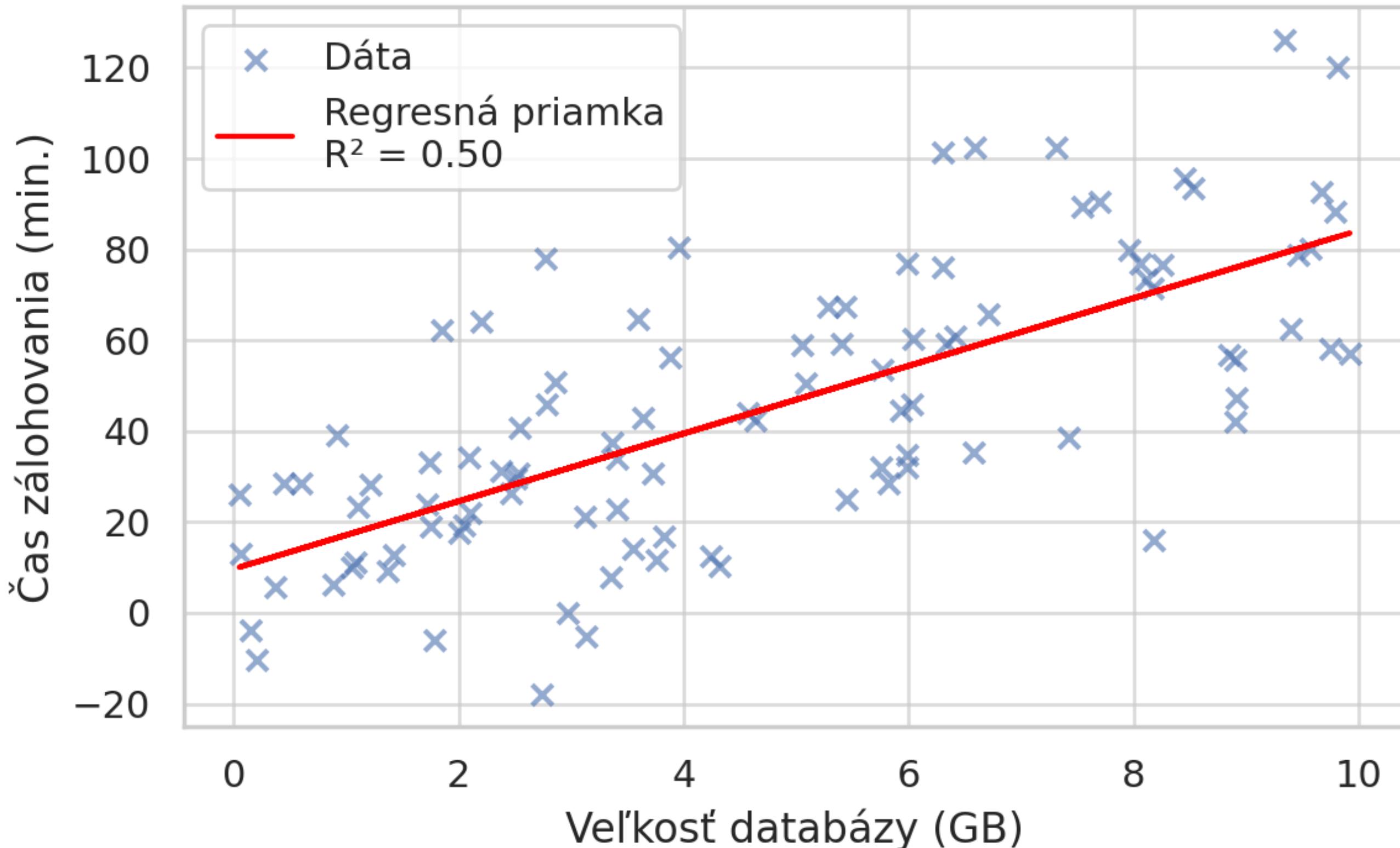
Marketing

-  **Počet newsletterov a miera odhlásenia**
 - Skúmanie, či častejšie zasielanie newsletterov zvyšuje mieru odhlásenia
 - Ak $R^2 \approx 0,65$ → viac e-mailov vede k väčšiemu odlivu odberateľov
-  **Zľava (%) a objem predaja**
 - Analýza, či vyššia zľava vede k vyššiemu predaju v kusoch
 - Ak $R^2 \approx 0,9$ → silná lineárna závislosť

Reklamné výdavky vs. počet zákazníkov ($R^2 \approx 0.7$)



Veľkosť databázy vs. čas zálohovania ($R^2 \approx 0.85$)



Regresný Koeficient Determin. R^2

Vzdelávanie

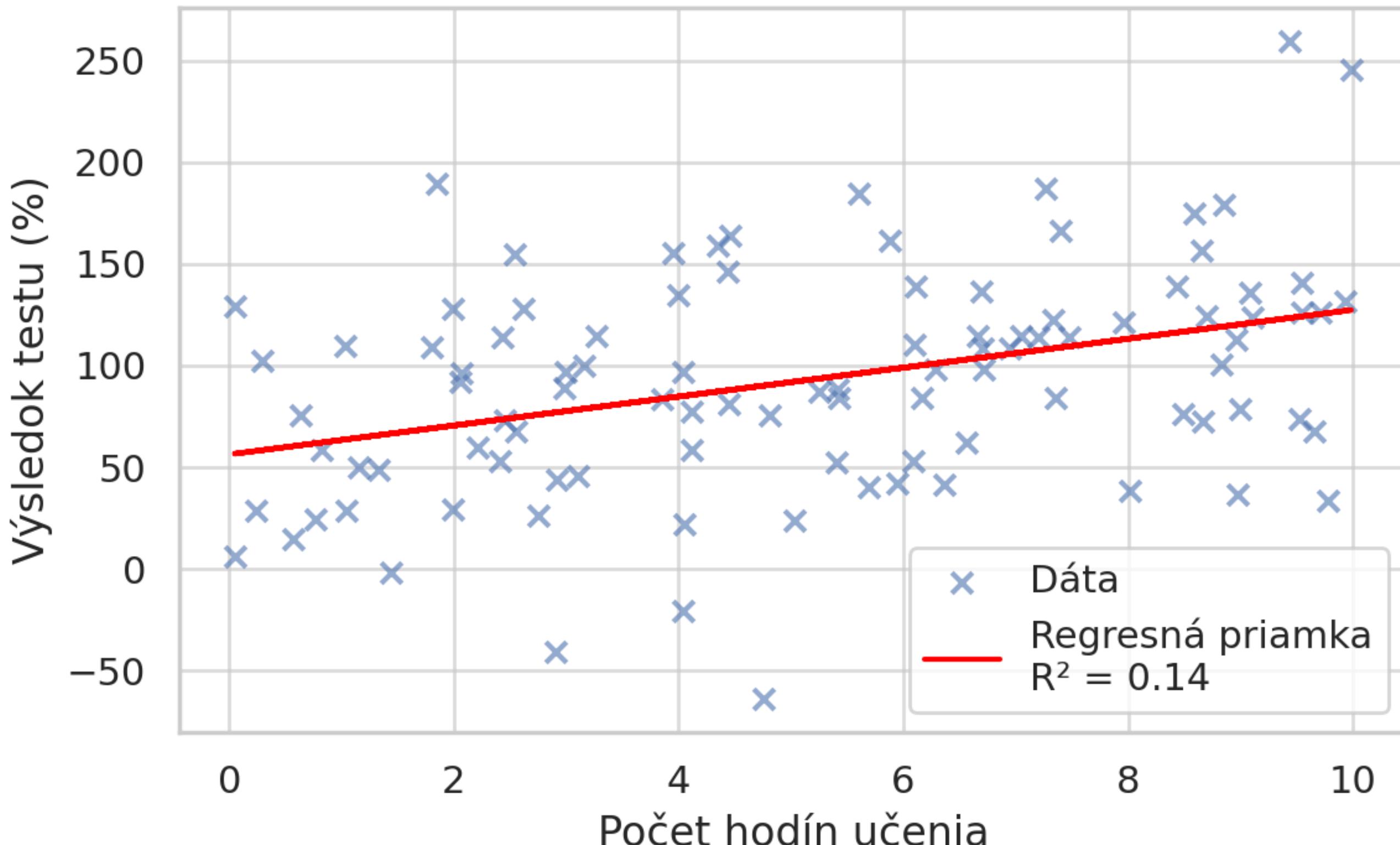
- Počet hodín učenia a výsledná známka**
 - Analýza, či viac učenia vedie k lepšiemu hodnoteniu
 - Ak $R^2 \approx 0,7$ → študenti, ktorí sa viac učia, dosahujú lepšie výsledky

- Vek študentov a výkonnosť v online kurzoch**
 - Skúmanie, či starší študenti dosahujú iné výsledky ako mladší
 - Ak $R^2 \approx 0,3$ → slabá závislosť

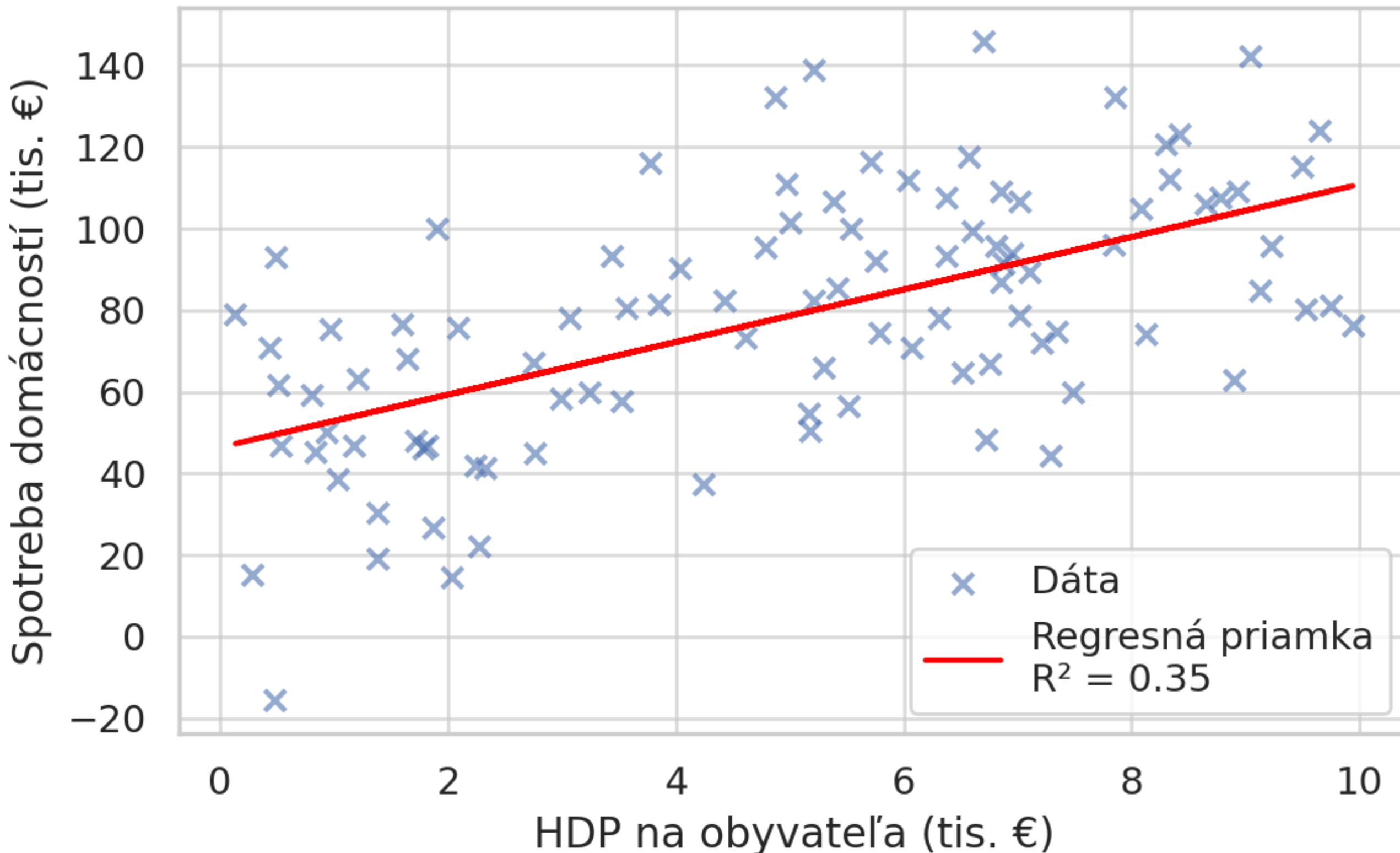
Ekonomika a financie

- HDP a spotreba domácností**
 - Analýza, ako rast HDP ovplyvňuje spotrebú domácností
 - Ak $R^2 \approx 0,75$ → vyšší HDP vedie k vyššej spotrebe
- Úroková sadzba a dopyt po hypotékach**
 - Skúmanie, či vyššia úroková sadzba znižuje počet žiadostí o hypotéky
 - Ak $R^2 \approx 0,6$ → mierna negatívna lineárna závislosť

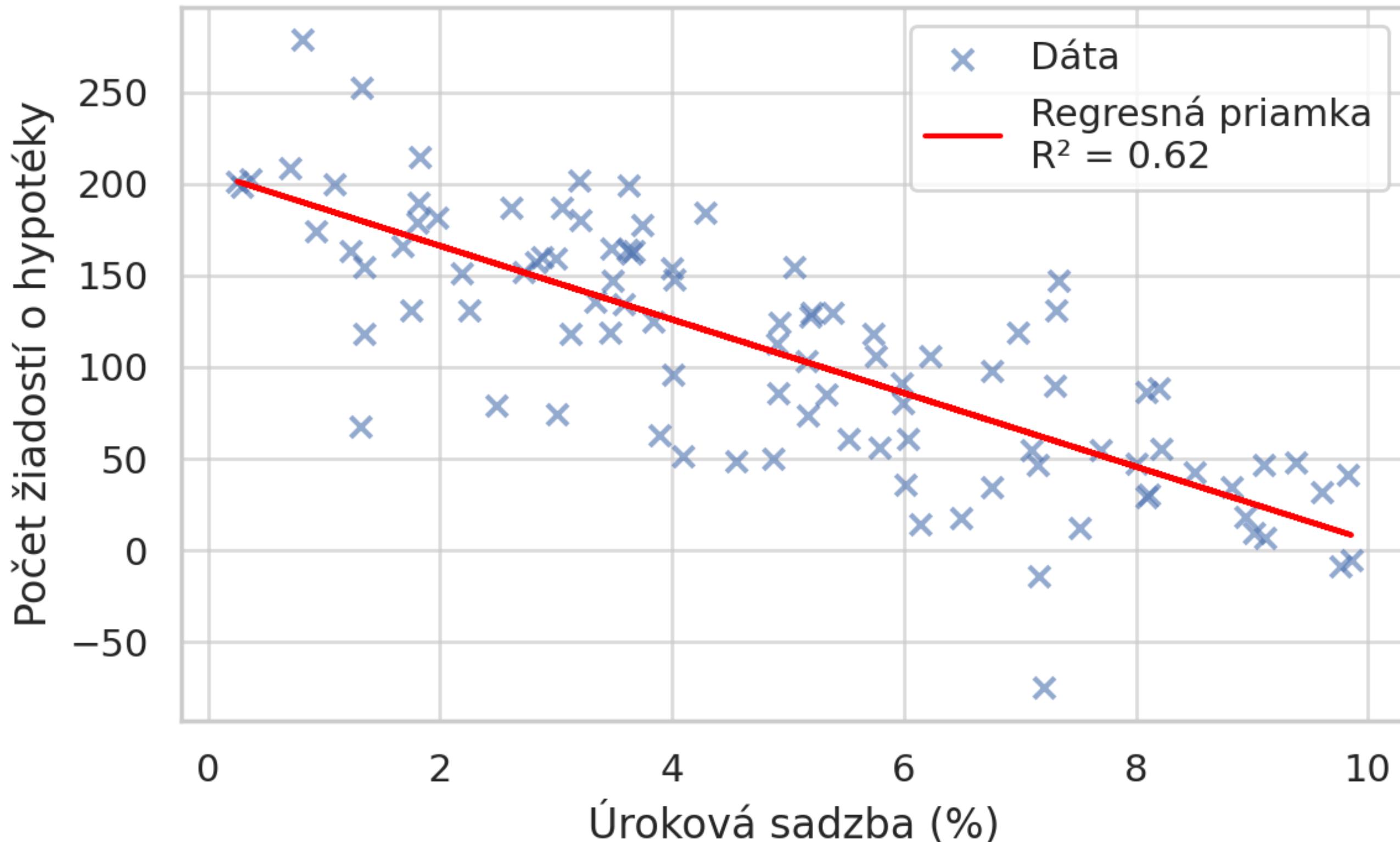
Počet hodín učenia vs. výsledok testu ($R^2 \approx 0.5$)



HDP vs. spotreba domácností ($R^2 \approx 0.75$)



Úroková sadzba vs. dopyt po hypotékach ($R^2 \approx 0.6$)



Regresný Koeficient Determin.

R^2

- Manažment

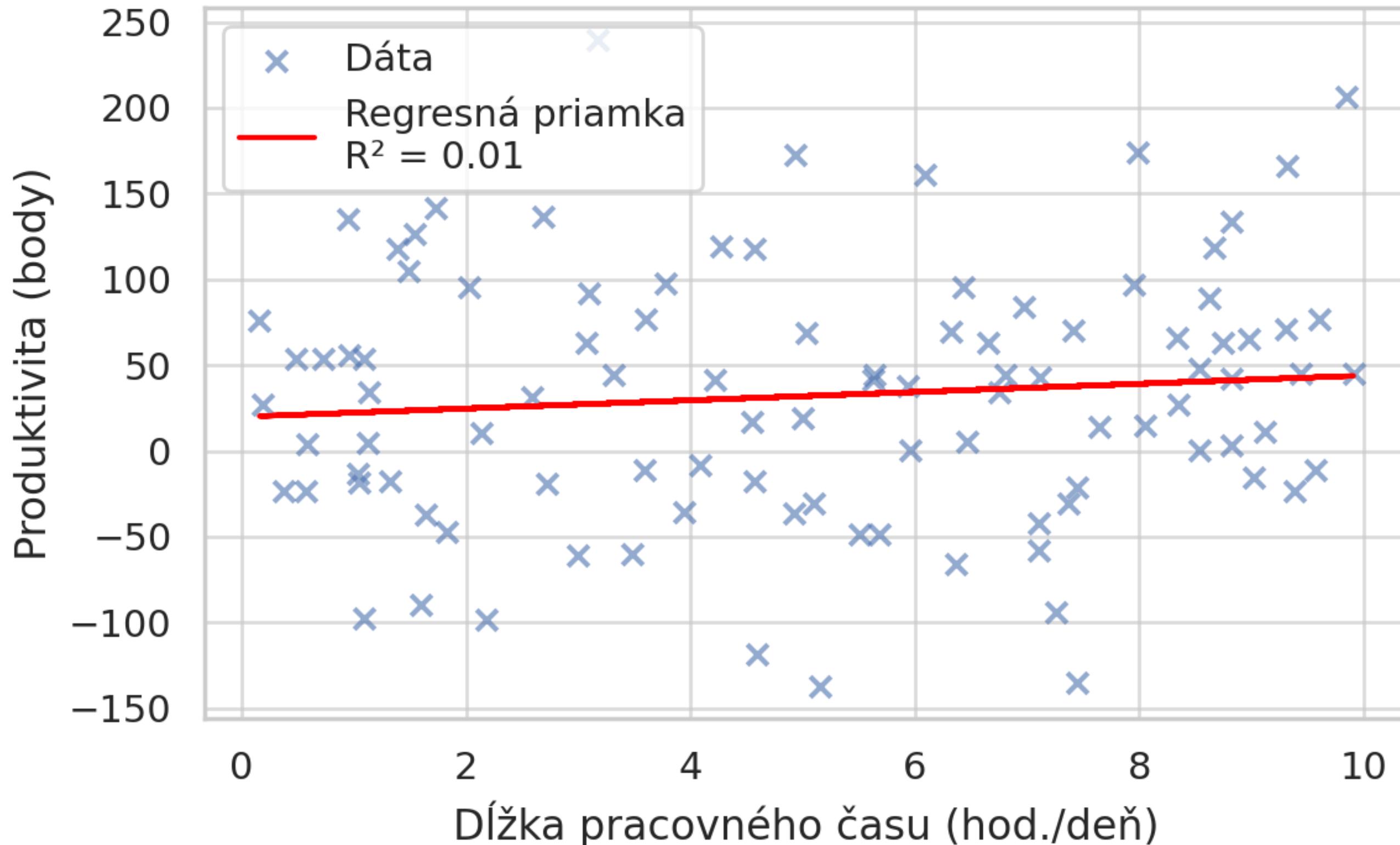
- ✓ Počet porád a efektivita rozhodovania

- Skúmanie, či viac porád viedie k lepším rozhodnutiam
 - Ak $R^2 \approx 0,3$ → slabá až mierna závislosť (pri prveľa poradách nastáva zdržanie)

- ✓ Fluktuácia zamestnancov a spokojnosť s vedením

- Analýza vplyvu spokojnosti s vedením na mieru odchodov zamestnancov
 - Ak $R^2 \approx 0,6$ → silnejšia negatívna regresná závislosť (nízka spokojnosť viedie k vyššej fluktuácii)

Produktivita vs. dĺžka pracovného času ($R^2 \approx 0.2$)



Predikcia nových hodnôt



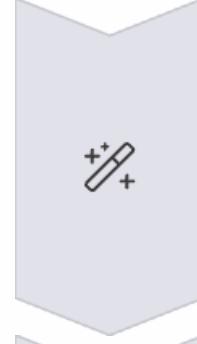
Natrénovaný model

Máme model natrénovaný na historických dátach. Kvalita predikcie závisí od vhodnosti modelu a kvality trénovacích dát. Pre lineárnu regresiu je model reprezentovaný koeficientmi priamky.



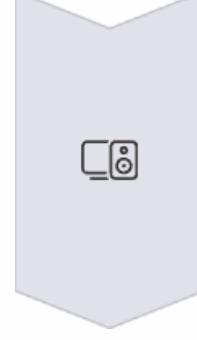
Nové vstupy

Pripravíme nové vstupné dáta v správnom formáte. Nové vstupy musia byť v rovnakom formáte ako trénovacie dáta, zvyčajne v tvare dvojrozmerného poľa (napr. [[1200]] pre jednu hodnotu alebo [[1000], [1500]] pre viacero).



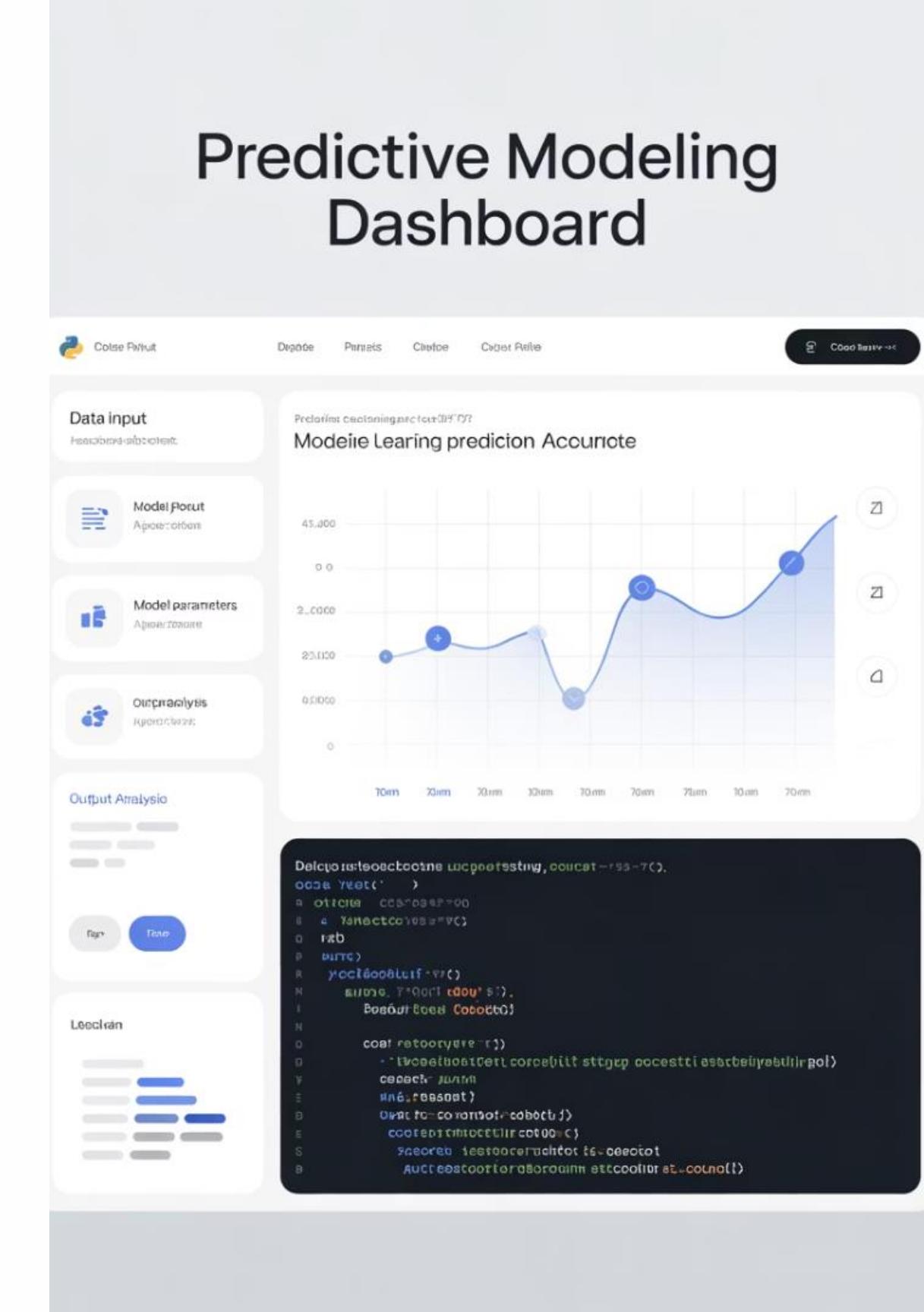
Predikcia

Použijeme metódu predict() na získanie predikcie. Táto metóda aplikuje natrénovaný algoritmus na nové dáta bez potreby opäťovného trénovania. Výsledkom je pole predikovaných hodnôt zodpovedajúce vstupom.



Interpretácia výsledkov

Analyzujeme a interpretujeme získané predikcie. Môžeme porovnať výsledky s očakávanými hodnotami, vizualizovať predikcie v grafe, alebo použiť metriky ako MAE, MSE či R² na vyhodnotenie presnosti modelu v reálnych podmienkach.



Multikolinearita v regresii

Čo je multikolinearita?

Multikolinearita nastáva, keď sú prediktorové premenné silno korelované medzi sebou. To môže spôsobiť nestabilitu koeficientov a stážiť interpretáciu modelu.

Poznámka: Nestabilita sa prejavuje veľkými zmenami v odhadovaných koeficientoch pri malých zmenách v dátach. Príkladom je situácia, keď máme dve premenné "výška" a "váha", ktoré sú prirodzene korelované.

Detekcia multikolinearity

1. Korelačná matica: sns.heatmap(X.corr(), annot=True)

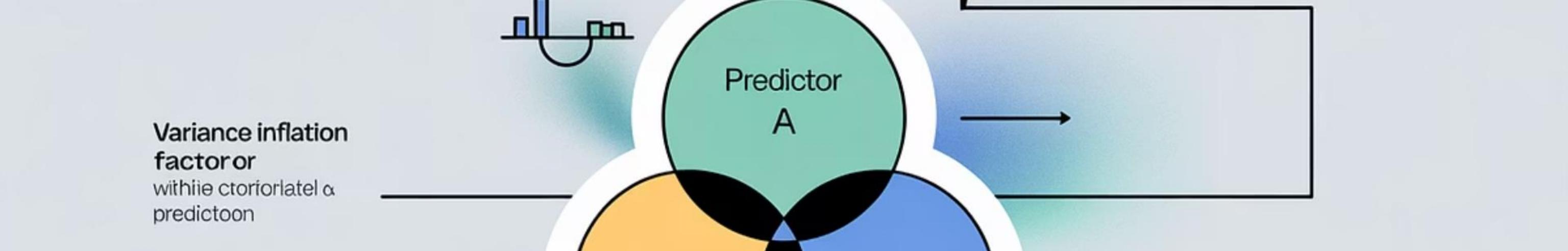
Poznámka: Hodnoty blízke k 1 alebo -1 medzi prediktormi naznačujú potenciálny problém. Korelačné hodnoty >0.7 alebo <-0.7 sú už považované za vysoké.

2. Variance Inflation Factor (VIF):

```
from statsmodels.stats.outliers_influence import variance_inflation_factor  
  
vif_data = pd.DataFrame()  
  
vif_data["Feature"] = X.columns  
  
vif_data["VIF"] = [variance_inflation_factor(X.values, i) for i in range(X.shape[1])]  
  
print(vif_data)
```

Poznámka: VIF >10 zvyčajne indikuje problém s multikolinearitou. VIF presne vyjadruje, kol'kokrát je variancia koeficientu zvýšená v dôsledku multikolinearity v porovnaní so situáciou, keď by premenné neboli korelované.

Praktická poznámka: Multikolinearita neznižuje prediktívnu silu modelu ako celku, ale robí individuálny vplyv prediktorov ľahšie interpretovateľným a znižuje štatistickú významnosť koeficientov.



Variance inflation factor
withiie ctoforlatel α
predictoon

Riešenie multikolinearity



Odstránenie premenných

Identifikácia a odstránenie jednej z vysoko korelovaných premenných.

Poznámka: Používame korelačnú maticu na identifikáciu párov s koreláciou > 0.7 . Odstránime tú premennú, ktorá má menší význam pre model alebo nižšiu koreláciu s cieľovou premennou.



Regularizácia

Použitie Ridge alebo Lasso regresie na zníženie vplyvu multikolinearity.

Poznámka: Ridge (L2) regularizácia zmenšuje všetky koeficienty, zatiaľ čo Lasso (L1) môže nastaviť niektoré koeficienty presne na nulu, čím vykonáva aj výber premenných.

3

Zníženie rozmernosti

Použitie PCA na vytvorenie nekorelovaných hlavných komponentov.

Poznámka: PCA transformuje pôvodné premenné na nové, nekorelované komponenty. Nevýhodou je strata interpretateľnosti pôvodných premenných.



Vytvorenie kompozitných premenných

Kombinácia korelovaných premenných do jednej novej premennej.

Poznámka: Napríklad zlúčením premenných výška a hmotnosť do BMI, alebo vytvorením indexu, ktorý kombinuje súvisiace ekonomicke ukazovatele.

Regularizácia v regresii

Regularizačné techniky pomáhajú predchádzať preučeniu (overfitting) a zlepšujú generalizáciu modelu.

Lineárna regresia

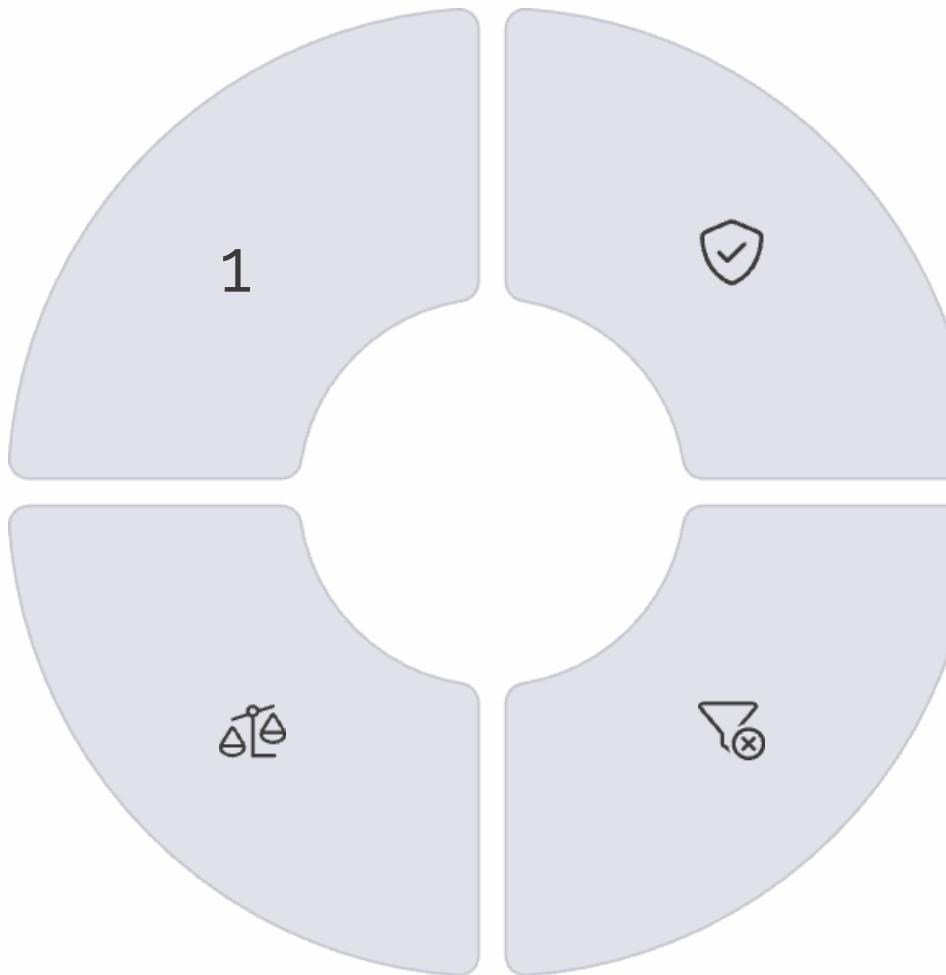
Štandardná regresia bez regularizácie,
minimalizuje len MSE.

Môže viest' k preučeniu pri
multikolinearite alebo
vysokorozmerných dátach.

ElasticNet

Kombinuje L1 a L2 regularizáciu pre
vyvážený prístup.

Spája výhody oboch metód - selekciu
premenných z Lasso a stabilitu Ridge
regresie.



Ridge regresia (L2)

Pridáva penalizáciu na základe
druhých mocnín koeficientov.

Redukuje váhy všetkých premenných,
ale žiadne nevynuluje. Vhodná pri
multikolinearite.

Lasso regresia (L1)

Pridáva penalizáciu na základe
absolútnej hodnoty koeficientov.

Dokáže vynulovať koeficienty menej
dôležitých premenných, čím vykonáva
aj selekciu premenných.

Volba regularizačnej metódy závisí od konkrétneho problému, štruktúry dát a cieľov modelovania.

Ridge regresia (L₂ regularizácia)

Princíp Ridge regresie

Ridge regresia pridáva k štandardnej chybovej funkcií (MSE) penalizačný člen založený na druhých mocninách koeficientov:

$$L = \text{MSE} + \alpha \times \sum(\beta_i^2)$$

kde α je regularizačný parameter, ktorý kontroluje silu penalizácie.

Poznámka: Ridge regresia je vhodná, keď máme multikolinearitu v dátach. Na rozdiel od Lasso, nikdy úplne nevynuluje koeficienty, len ich zmenšuje.

Poznámka: L₂ regularizácia má tendenciu distribuovať váhy rovnomernejšie medzi korelované premenné, čo vedie k robustnejšiemu modelu.

Implementácia v scikit-learn

```
from sklearn.linear_model import Ridge  
  
# Vytvoríme Ridge model s parametrom alpha=1.0  
ridge_model = Ridge(alpha=1.0)  
  
# Trénujeme model na škálovaných dátach  
ridge_model.fit(X_train_scaled, y_train)  
  
# Predikcia na testovacej množine  
y_pred_ridge = ridge_model.predict(X_test_scaled)  
  
# Vyhodnotenie presnosti modelu  
r2_ridge = r2_score(y_test, y_pred_ridge)  
  
print("R2 (Ridge):", r2_ridge)
```

Poznámka: Parameter alpha kontroluje intenzitu regularizácie. Väčšie hodnoty alpha spôsobujú väčšiu penalizáciu a vedú k menším koeficientom.

Poznámka: Škálovanie vstupných dát je pri Ridge regresii dôležité, pretože regularizácia je citlivá na merné jednotky premenných.

Lasso regresia (L1 regularizácia)

Princíp Lasso regresie

Lasso regresia pridáva k štandardnej chybovej funkcií (MSE) penalizačný člen založený na absolútnych hodnotách koeficientov:

$$L = \text{MSE} + \alpha \times \sum |\beta_i|$$

kde α je regularizačný parameter, ktorý kontroluje silu penalizácie.

Poznámka: Na rozdiel od Ridge regresie, Lasso dokáže úplne vynulovať niektoré koeficienty, čím vykonáva automatický výber premenných.

Vlastnosti: Lasso je vhodné použiť, keď predpokladáme, že len malá časť vstupných premenných má vplyv na výsledok (tzv. sparse model).

Implementácia v scikit-learn

```
from sklearn.linear_model import Lasso  
lasso_model = Lasso(alpha=0.1)  
lasso_model.fit(X_train_scaled, y_train)  
y_pred_lasso = lasso_model.predict(X_test_scaled)  
r2_lasso = r2_score(y_test, y_pred_lasso)  
print("R2 (Lasso):", r2_lasso)
```

Poznámka: Parameter α je klúčový - vyššie hodnoty znamenajú väčšiu regularizáciu a viac nulových koeficientov.

Tip: Pred použitím Lasso regresie je dôležité štandardizovať vstupné premenné, inak by penalizácia závisela od merných jednotiek.

ElasticNet (kombinácia L1 a L2)

Princíp ElasticNet

ElasticNet kombinuje L1 a L2 regularizáciu:

$$L = \text{MSE} + \alpha \times (\rho \times \sum |\beta_i| + (1-\rho) \times \sum (\beta_i^2)/2)$$

kde α je celková sila regularizácie a ρ je pomer medzi L1 a L2 regularizáciou (`l1_ratio`).

Poznámka: *ElasticNet prekonáva nedostatky samotných L1 a L2 metód. Je obzvlášť užitočný pri modelovaní s mnohými korelovanými premennými.*

Poznámka: *ElasticNet je vhodný najmä pre datasety s veľkým počtom prediktorov, kde niektoré z nich môžu byť korelované. Kombinuje schopnosť Lasso odstraňovať nepotrebné premenné a schopnosť Ridge regresie pracovať so skupinami korelovaných premenných.*

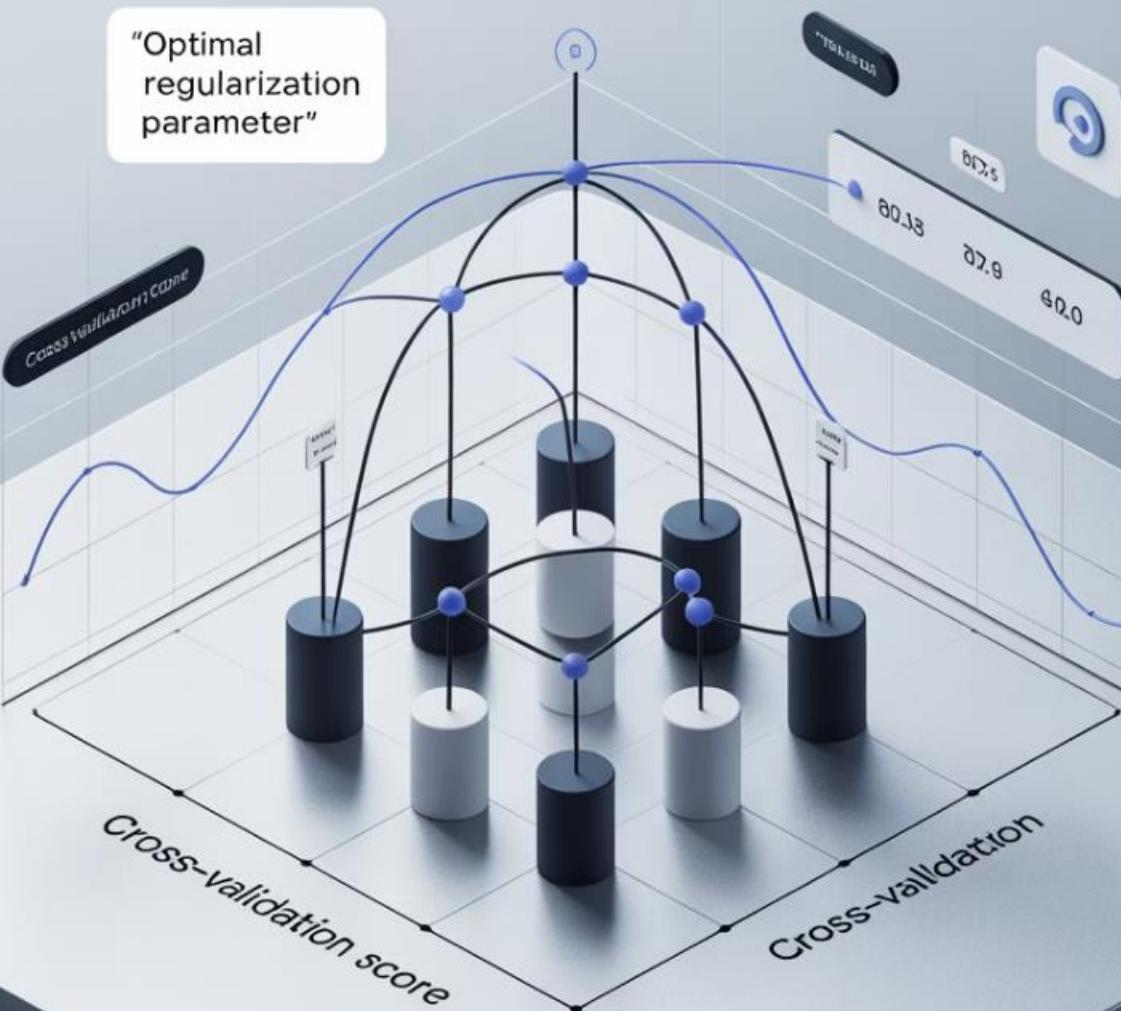
Implementácia v scikit-learn

```
from sklearn.linear_model import ElasticNet  
  
elastic_model = ElasticNet(alpha=0.1, l1_ratio=0.5)  
  
elastic_model.fit(X_train_scaled, y_train)  
  
y_pred_elastic = elastic_model.predict(X_test_scaled)  
  
r2_elastic = r2_score(y_test, y_pred_elastic)  
  
print("R2 (ElasticNet):", r2_elastic)
```

Poznámka: Parameter `l1_ratio=0.5` znamená rovnováhu medzi L1 a L2. Pri hodnote 1 sa *ElasticNet* správa ako *Lasso*, pri hodnote 0 ako *Ridge* regresia.

Hyperparameter Tuning

Mepemameer
Learning



Výber optimálneho regularizačného parametra



Definícia rozsahu parametrov

Vytvorenie zoznamu možných hodnôt alpha.

Typicky používame logaritmickú škálu (napr. 0.001, 0.01, 0.1, 1, 10, 100) na efektívne pokrytie širokého rozsahu hodnôt.

Krízová validácia

Použitie GridSearchCV alebo RandomizedSearchCV na nájdenie optimálnej hodnoty.

GridSearchCV systematicky testuje všetky kombinácie, zatiaľ čo RandomizedSearchCV náhodne vzorkuje parameter space, čo je efektívnejšie pre veľké množstvo parametrov.

Vyhodnotenie výsledkov

Analýza výsledkov krízovej validácie a výber najlepšieho modelu.

Okrem najlepšieho skóre by sme mali analyzovať aj stabilitu výkonu medzi jednotlivými foldami, aby sme predišli preučeniu na konkrétnej podmnožine dát.

Tréning finálneho modelu

Použitie optimálneho parametra na tréning finálneho modelu.

Po nájdení najlepšieho parametra je dobré natrénovať model znova na celej trénovacej množine a následne vyhodnotiť jeho výkon na testovacej množine pre konečné posúdenie generalizácie.

Krížová validácia pre výber parametra

Implementácia GridSearchCV

```
from sklearn.model_selection import GridSearchCV  
  
from sklearn.linear_model import Ridge  
  
// Definujeme rozsah hodnôt parametra alpha, ktorý chceme otestovať  
param_grid = {'alpha': [0.001, 0.01, 0.1, 1, 10, 100]}  
  
// Inicializujeme základný Ridge regresný model  
ridge = Ridge()  
  
// GridSearchCV systematicky vyskúša všetky kombinácie parametrov  
// cv=5 znamená 5-násobnú krížovú validáciu  
  
grid_search = GridSearchCV(ridge, param_grid, cv=5, scoring='r2')  
  
// Trénujeme model na všetkých kombináciách parametrov  
grid_search.fit(X_train_scaled, y_train)  
  
// Vypíšeme najlepšie parametre a skóre  
  
print("Najlepší parameter:", grid_search.best_params_)  
  
print("Najlepšie skóre:", grid_search.best_score_)
```

Vizualizácia výsledkov

```
// Prevedieme výsledky krížovej validácie do DataFrame pre ľahšiu manipuláciu  
results = pd.DataFrame(grid_search.cv_results_)  
  
// Nastavíme veľkosť grafu pre lepšiu čitateľnosť  
plt.figure(figsize=(10, 6))  
  
// Vykreslíme priemery skóre s chybovými úsečkami pre každú hodnotu alpha  
plt.errorbar(results['param_alpha'], results['mean_test_score'], yerr=results['std_test_score'])  
  
// Použijeme logaritmickú škálu pre os x kvôli širokému rozsahu hodnôt alpha  
plt.xscale('log')  
  
// Popíšeme osi grafu a pridáme nadpis  
plt.xlabel('Alpha (log scale)')  
plt.ylabel('R2 skóre')  
plt.title('Výsledky krížovej validácie pre Ridge regresiu')  
plt.grid(True)  
plt.show()  
  
// Graf nám umožňuje vizuálne identifikovať optimálnu hodnotu alpha, pri ktorej  
model dosahuje najvyššie R2 skóre
```



Čo sa naučíme?

1. Aké máme typy regresií a ako sa používajú v praxi?
2. Kedy sa používa polynomiálna regresia?
3. Čo je Ridge a Lasso regresia?
4. Aký je rozdiel medzi lineárhou a logistickou regresiou?
5. Ako si pripraviť dáta na regresiu?
6. Aký typ regresie je vhodný pre reálne problémy?
7. Profit

Ako používať Lineárnu Regresiu

AKREDITOVANÝ KURZ





Čo sa naučíme?

1. Ako začať používať lineárnu regresiu?
2. Z akých častí sa skladá lineárny regresný model?
3. Ako sa model trénuje (fituje) v scikit-learn?
4. Ako interpretovať výstupy modelu (koeficienty a intercept)?
5. Ako sa vyhodnocuje presnosť lineárneho modelu?
6. Aké sú výhody a nevýhody lineárnej regresie?
7. Profit

Teória lineárnej regresie

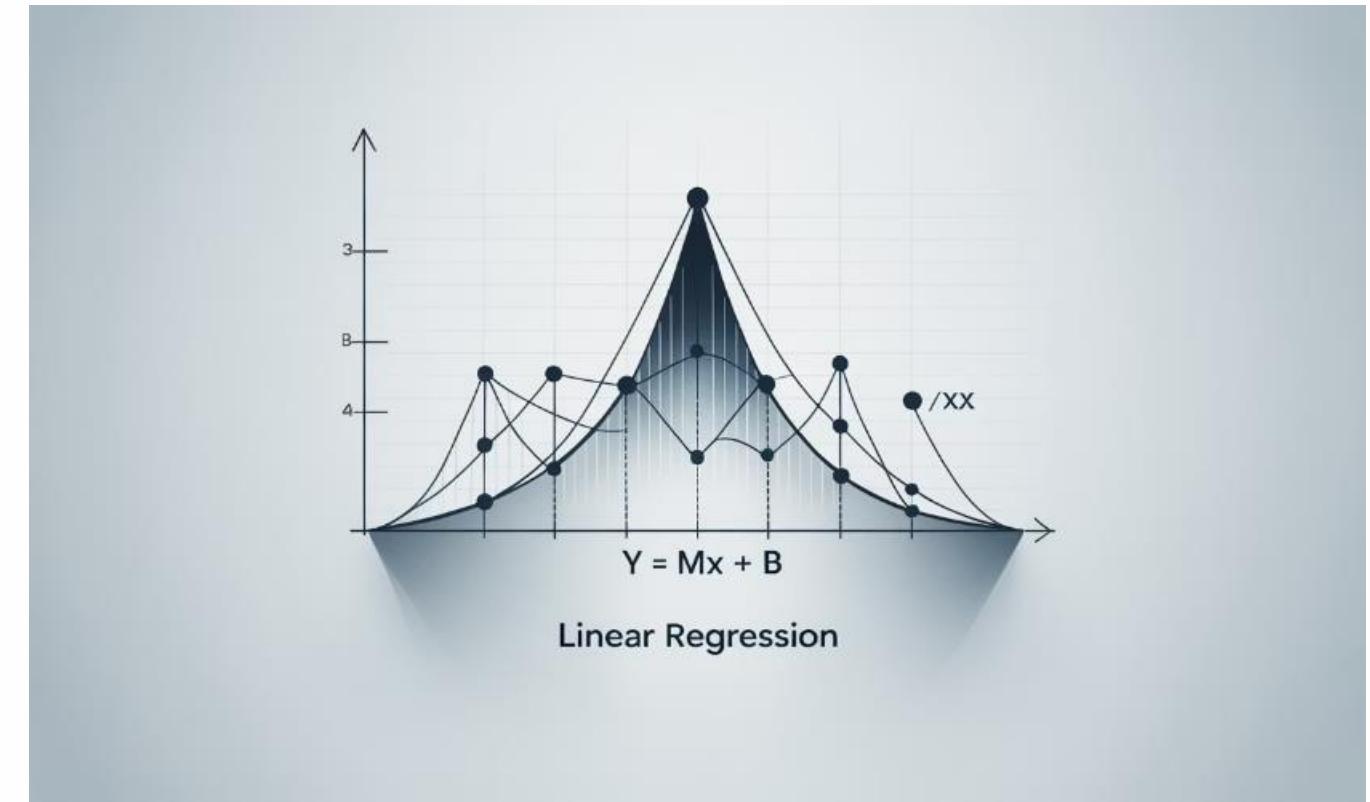
Matematický model

Lineárna regresia modeluje vzťah medzi závislou premennou y a jednou alebo viacerými nezávislými premennými X pomocou lineárnej funkcie:

$$y = \beta_0 + \beta_1 x + \epsilon$$

kde β_0 je intercept (priesečník), β_1 je sklon priamky (váha atribútu) a ϵ je chyba predikcie.

Poznámka: Tento model je základom pre pochopenie vzťahov medzi dátami a predikciu hodnôt na základe historických údajov.



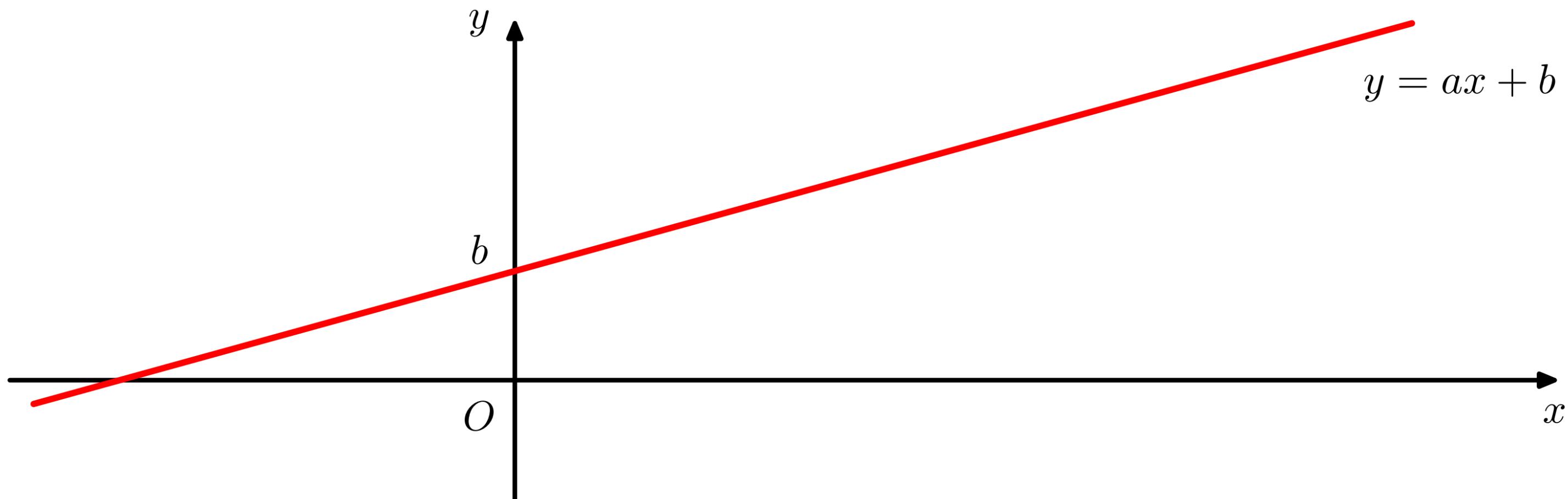
Poznámka: Vizualizácia ukazuje geometrickú interpretáciu lineárnej regresie, kde priamka predstavuje model snažiaci sa minimalizovať vzdialenosť medzi skutočnými bodmi a predikciami.

Poznámka: Lineárna regresia je najjednoduchší a najčastejšie používaný algoritmus strojového učenia, ktorý slúži ako základ pre zložitejšie modely. Pre optimálne výsledky je dôležité, aby dáta splňali určité predpoklady ako linearita, nezávislosť pozorovaní a normálne rozdelenie chýb.

Lineárna funkcia –
$$f : y = ax + b, \quad a, b \in \mathbb{R}, \quad a \neq 0$$

$$\mathcal{D}(f) = \mathbb{R}, \quad \mathcal{H}(f) = \mathbb{R}.$$

Grafom je priamka so smernicou a , ktorá na osi y vytína úsek b .



Matematický Model Lineárnej Regresie

1. **Y: Závislá premenná (predikovaná)**

2. **X: Nezávislá premenná (prediktor)**

3. **β_0 : Intercept (priesečník s osou Y)**

4. **β_1 : Koeficient regresie (slope)**

5. **ϵ : Chyba modelu**

$$Y = \beta_0 + \beta_1 X + \epsilon$$

$$y = a_0 + a_1 x,$$

Regresné Koeficienty

Intercept (β_0)

- **Hodnota**, ktorú **závislá premenná (Y) nadobúda**, keď je **hodnota nezávislej premennej (X) rovná 0**.
 - Inými slovami, intercept je **bod, kde regresná priamka pretína os Y**.
 - Matematicky: $Y = \beta_0$, keď $X = 0$.
- **Interpretácia:**
Predstavuje počiatočnú hodnotu závislej premennej, keď nezávislá premenná nemá žiadny vplyv (napríklad cena produktu pri nulovej hodnote predaja).

Slope (β_1)

- **Koeficient**, ktorý určuje, **ako veľmi sa zmení hodnota závislej premennej (Y) pri zmene nezávislej premennej (X) o jednotku**.
 - Matematicky: Slope (β_1) udáva **zmenu hodnoty Y** pri zmene X o 1.
 - Predstavuje **sklon priamky**: ak je slope pozitívny, priamka stúpa, ak je negatívny, priamka klesá.
- **Interpretácia:**
Ak slope (β_1) je 2, znamená to, že pri zvýšení hodnoty X o 1 sa hodnota Y zvýši o 2. Ak je β_1 -3, znamená to, že pri zvýšení hodnoty X o 1 sa hodnota Y zníži o 3.

Regresná Rovnica

- Regresná rovnica môže vyzerat' napr. takto: $Y = 5+2X$
- **A. Intercept ($\beta_0 = 5$):** Ak $X = 0$, potom $Y = 5$ (počiatočná hodnota Y).
- **B. Slope ($\beta_1 = 2$):** Každé zvýšenie X o 1 spôsobí, že Y sa zvýši o 2.
- Tento model teda vyjadruje lineárny vztah medzi X a Y , kde intercept je 5 a slope je 2.

Ako Postupovať?

1. Identifikovať premenné

- Určiť závislú a nezávislé premenné.

2. Vytvoriť model

- Určiť, aký typ regresie je najvhodnejší.

3. Vyhodnotiť model

- Použiť metódy ako R^2 , F-test alebo p-hodnoty na testovanie účinnosti modelu.

4. Predikcia

- Použiť model na predpovedanie neznámych hodnôt závislej premennej.

Linear Regression: Predicting the Future



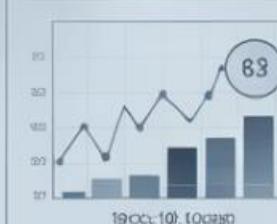
FINANCE



AUTOMOTIVE



REAL ESTATE



STOCK MARKET TRENDES

DATA SOURCE: BLOOMBERG
DATE SOURCE: BLOOMBERG

VEALE SALES PROOCLING

DATA SOURCE: BLOOMBERG
DATE SOURCE: BLOOMBERG

Aplikácie lineárnej regresie



Finančné predpovede

Predikcia výšky mzdy na základe rokov praxe, odhad výnosov investícií, predpoved' tržieb.



Automobilový priemysel

Predikcia spotreby paliva podľa hmotnosti vozidla, odhad životnosti komponentov.



Analýza trendov

Identifikácia a extrapolácia trendov v časových radoch, predpoved' budúceho vývoja.



Realitný trh

Odhad ceny nehnuteľnosti na základe jej parametrov, predikcia výšky hypotéky.

Tthototun(+)
oooy-t-Tote:]-!(+)-i-()i(+>(+),
pptuiul|@l|,-[]it()it(+)
ton., (i]er-|tateet(-|te-jth(-t)
a tots, (-[tcleelv@(-jscoutelTj((z)
5 opiilsunt.-|edi-litjgvihter())|(+>
a adtunett teatr | det-caxtdnescoetesitronata(+)

Implementácia lineárnej regresie v scikit-learn

Import knižníc

```
from sklearn.linear_model import  
LinearRegression
```

Najprv importujeme triedu LinearRegression z balíka scikit-learn, ktorá poskytuje implementáciu metódy najmenších štvorcov.

Vytvorenie modelu

```
model = LinearRegression()
```

Vytvoríme inštanciu lineárneho regresného modelu. Môžeme špecifikovať parametre ako fit_intercept=True/False podľa potreby.

Trénovanie modelu

```
model.fit(X_train, y_train)
```

Natrénujeme model na trénovacích dátach, kde X_train obsahuje vlastnosti (features) a y_train cieľové hodnoty. Metóda fit() vypočíta optimálne koeficienty.

Získanie koeficientov

```
model.coef_, model.intercept_
```

Po natrénovaní môžeme získať koeficienty (sklony) ako model.coef_ a posun (intercept) ako model.intercept_. Tieto hodnoty definujú našu regresnú priamku.

Vizualizácia lineárnej regresie

Príprava dát

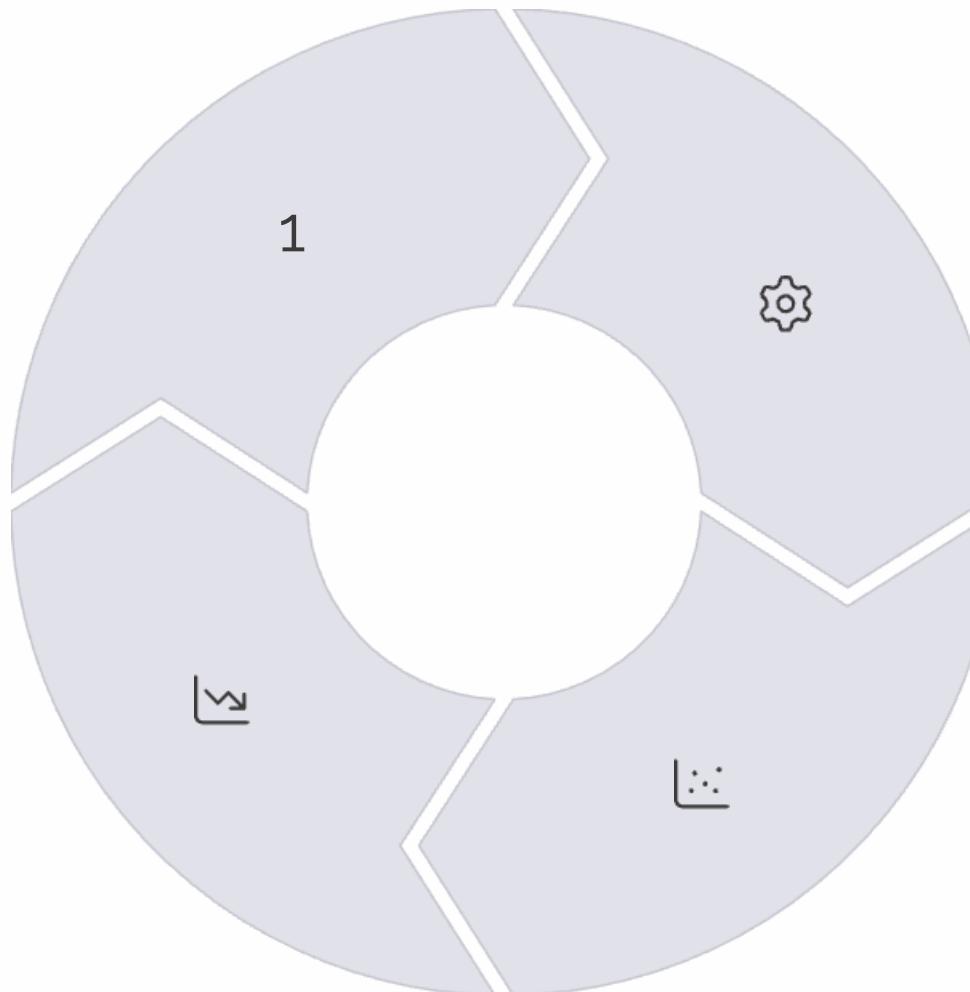
Načítanie a rozdelenie dát na trénovaciu a testovaciu množinu.

```
import pandas as pd from sklearn.model_selection  
import train_test_split data =  
pd.read_csv('data.csv') X = data[['feature']] y =  
data['target'] X_train, X_test, y_train, y_test =  
train_test_split(X, y, test_size=0.2,  
random_state=42)
```

Vykreslenie regresnej priamky

Pridanie predikovaných hodnôt ako regresnej priamky.

```
import numpy as np y_pred = model.predict(X_test)  
# Zoradenie pre správne vykreslenie čiary sort_idx =  
np.argsort(X_test.flatten())  
plt.plot(X_test.iloc[sort_idx], y_pred[sort_idx],  
color='red', linewidth=2, label='Regresná priamka')  
plt.legend() plt.savefig('linear_regression.png')  
plt.show()
```



Trénovanie modelu

Vytvorenie a trénovanie lineárneho regresného modelu.

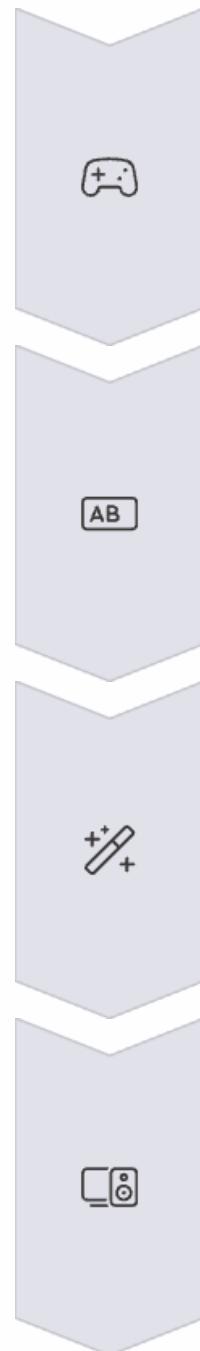
```
from sklearn.linear_model import LinearRegression  
model = LinearRegression() model.fit(X_train,  
y_train) # Koeficienty modelu print(f"Koeficient:  
{model.coef_[0]:.4f}") print(f"Intercept:  
{model.intercept_:.4f}")
```

Vykreslenie bodov

Zobrazenie skutočných hodnôt ako bodov v grafe.

```
import matplotlib.pyplot as plt  
plt.figure(figsize=(10, 6)) plt.scatter(X_test, y_test,  
color='blue', label='Skutočné hodnoty')  
plt.xlabel('Nezávislá premenná') plt.ylabel('Závislá  
premenná') plt.title('Lineárna regresia')
```

Predikcia nových hodnôt



Natrénovaný model

Máme model natrénovaný na historických dátach. Kvalita predikcie závisí od vhodnosti modelu a kvality trénovacích dát. Pre lineárnu regresiu je model reprezentovaný koeficientmi priamky.

Nové vstupy

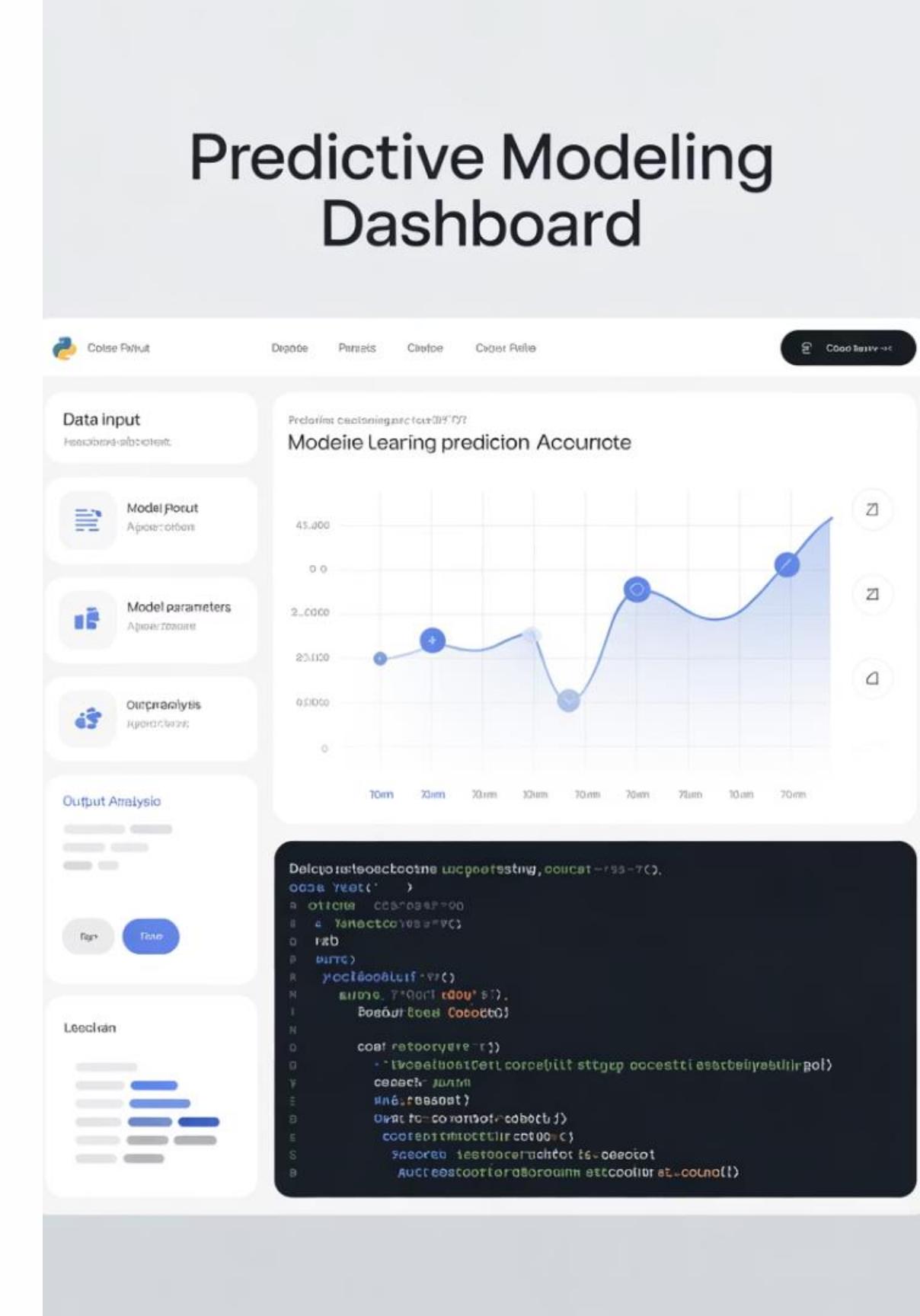
Pripravíme nové vstupné dáta v správnom formáte. Nové vstupy musia byť v rovnakom formáte ako trénovacie dáta, zvyčajne v tvare dvojrozmerného poľa (napr. [[1200]] pre jednu hodnotu alebo [[1000], [1500]] pre viacero).

Predikcia

Použijeme metódu `predict()` na získanie predikcie. Táto metóda aplikuje natrénovaný algoritmus na nové dátá bez potreby opäťovného trénoania. Výsledkom je pole predikovaných hodnôt zodpovedajúce vstupom.

Interpretácia výsledkov

Analyzujeme a interpretujeme získané predikcie. Môžeme porovnať výsledky s očakávanými hodnotami, vizualizovať predikcie v grafe, alebo použiť metriky ako MAE, MSE či R^2 na vyhodnotenie presnosti modelu v reálnych podmienkach.



Príklad predikcie nových hodnôt

Kód pre predikciu

Po natrénovaní modelu môžeme predpovedať hodnoty pre nové vstupy:

```
novy_vstup = [[1200]] # Dvojité zátvorky pre 2D array formát  
predikcia = model.predict(novy_vstup) # Aplikácia modelu na vstupné dátá  
  
print(predikcia) # Výstup bude predikovaná hodnota pre vstup 1200
```

Poznámka: Model očakáva 2D array, preto používame dvojité zátvorky aj pri jednom vstupe.

Poznámka: Pri predikcii vždy skontrolujte, či sú vstupné dátá v rovnakom formáte a rozsahu ako trénovacie dátá. Prípadne použite rovnaké transformácie, ktoré boli aplikované na trénovacie dátá.

Predikcia pre viaceré vstupy

Môžeme predpovedať hodnoty pre viacero vstupov naraz:

```
vstupy = [[1000], [1500], [1800]] # Zoznam viacerých vstupov  
predikcie = model.predict(vstupy) # Dávková predikcia  
  
print(predikcie) # Výstup bude pole predikovaných hodnôt
```

Poznámka: Dávkové predikcie sú efektívnejšie ako postupné predikcie jednotlivých hodnôt.

Jednoduchá lineárna regresia - model s 1 premenou

Ciel'

Vybudovať a vyhodnotiť model jednoduchej lineárnej regresie pomocou jednej vstupnej premennej MedInc (medián príjmu) na predikciu ceny domov.

Dataset California Housing

Dataset obsahuje informácie o cenách domov v Kalifornii spolu s rôznymi atribútmi ako medián príjmu, vek domov, počet izieb a ďalšie. Pre jednoduchú regresiu vyberieme len jeden atribút - medián príjmu (MedInc).

Výber jednej vstupnej premennej



Import knižníc a načítanie dát

```
from sklearn.datasets import fetch_california_housing  
import pandas as pd  
data = fetch_california_housing()  
df = pd.DataFrame(data.data, columns=data.feature_names)  
df["target"] = data.target
```

Poznámka: Importujeme scikit-learn a pandas knižnice, ktoré sú klúčové pre dátovú analýzu. Následne načítame dataset kalifornských domov a vytvoríme pandas DataFrame pre jednoduchšiu manipuláciu s dátami.

Preskúmanie dát

```
print(df.head())  
print(df.describe())
```

Poznámka: Metóda head() zobrazí prvých 5 riadkov datasetu, čo nám umožní rýchly náhľad na štruktúru dát. Metóda describe() poskytuje základné štatistické ukazovatele (priemer, min, max, kvartily) pre všetky numerické stĺpce.

Výber premennej Medlnc

```
X = df[["Medlnc"]]  
y = df["target"]
```

Poznámka: Pre jednoduchú lineárnu regresiu vyberáme len jednu nezávislú premennú - Medlnc (medián príjmu). Dvojité hranasté zátvorky vytvárajú DataFrame, nie sériu, čo je požadovaný formát pre scikit-learn. Cieľová premenná (cena domu) je označená ako y.

Zobrazenie vzťahu

```
plt.scatter(df["Medlnc"], df["target"], alpha=0.3)  
plt.xlabel("Medián príjmu")  
plt.ylabel("Cena domu")  
plt.show()
```

Poznámka: Vizualizujeme vzťah medzi mediánom príjmu a cenou domu pomocou bodového grafu. Parameter alpha=0.3 nastavuje priehľadnosť bodov, čo je užitočné pri veľkom množstve prekrývajúcich sa bodov. Tento graf nám pomôže vizuálne posúdiť, či existuje lineárny vzťah medzi vybranými premennými pred vytvorením modelu.

Rozdelenie dát na trénovaciu a testovaciu množinu

Kód pre rozdelenie dát

```
from sklearn.model_selection import train_test_split  
  
X = df[["MedInc"]]  
  
y = df["target"]  
  
X_train, X_test, y_train, y_test = train_test_split(X, y,  
test_size=0.2, random_state=42)  
  
print("Tréningová množina:", X_train.shape)  
  
print("Testovacia množina:", X_test.shape)
```

Význam parametrov

- `test_size=0.2` - 20% dát bude použitých na testovanie, 80% na trénovanie
- `random_state=42` - zabezpečuje reprodukovateľnosť výsledkov

Rozdelenie dát je kľúčové pre objektívne vyhodnotenie modelu na nevidených dátach.

Tréning lineárneho regresného modelu

Import a inicializácia

```
</>  
from sklearn.linear_model import LinearRegression  
  
model = LinearRegression()
```

Importujeme triedu LinearRegression z knižnice scikit-learn a vytvoríme nový prázdný model, ktorý bude reprezentovať našu lineárnu regresnú funkciu.

Tréning modelu

```
model.fit(X_train, y_train)
```

Metóda fit() natrénuje model na trénovacej množine dát. Počas tohto procesu sa vypočíta optimálna hodnota koeficientu a interceptu tak, aby minimalizovali stratu (chybu predikcie).

Získanie koeficientov

```
print("Koeficient:", model.coef_[0])  
  
print("Intercept:", model.intercept_)
```

Výpis natrénovaných parametrov modelu. Koeficient (sklon priamky) vyjadruje, ako sa zmení cieľová premenná pri zmene vstupnej premennej o jednotku. Intercept predstavuje hodnotu cieľovej premennej, keď je vstupná premenná nulová.

Predikcia

```
y_pred = model.predict(X_test)
```

Aplikácia natrénovaného modelu na testovacie dáta pomocou metódy predict(). Vytvorí predikcie (odhady) pre cieľové hodnoty na základe vstupných premenných z testovacej množiny.

Lineárna regresia hľadá najlepšiu líniu, ktorá popisuje vzťah medzi vstupnými premennými a cieľovou premennou. Výsledkom tréningu je matematická rovnica v tvare $y = mx + b$, kde m je koeficient a b je intercept.



Vizualizácia regresnej priamky

Kód pre vizualizáciu

```
import matplotlib.pyplot as plt

# Poznámka: Matplotlib je základná knižnica pre vizualizáciu dát v Pythone
plt.figure(figsize=(8, 6))

# Poznámka: Nastavenie veľkosti grafu pre lepšiu čitateľnosť

plt.scatter(X_test, y_test, color='blue', alpha=0.3, label="Skutočné hodnoty")

# Poznámka: Bodový graf zobrazuje skutočné dáta - každý bod reprezentuje jeden dom

plt.plot(X_test, y_pred, color='red', label="Regresná priamka")

# Poznámka: Červená čiara reprezentuje predikcie modelu pre testovací dataset

plt.xlabel("MedInc (Medián príjmu)")

plt.ylabel("Cieľová premenná (MedHouseVal)")

# Poznámka: Popisky osí pomáhajú pochopiť, aké veličiny sledujeme

plt.title("Jednoduchá lineárna regresia: MedInc vs. cena domu")

# Poznámka: Názov grafu vystihuje podstatu vizualizácie

plt.legend()

# Poznámka: Legenda pomáha identifikovať význam farieb v grafe

plt.grid(True)

# Poznámka: Mriežka zlepšuje orientáciu v hodnotách

plt.show()

# Poznámka: Príkaz na zobrazenie finálneho grafu
```

Interpretácia grafu

Graf zobrazuje vzťah medzi mediánom príjmu (os x) a cenou domu (os y). Modré body predstavujú skutočné hodnoty, červená čiara je regresná priamka predikovaná modelom.

Poznámka: Rozptyl bodov okolo priamky ukazuje, ako dobre model vysvetľuje variabilitu dát

Sklon priamky zodpovedá koeficientu modelu a indikuje, o koľko sa zvýši cena domu pri zvýšení mediánu príjmu o jednotku.

Poznámka: Pozitívny sklon naznačuje pozitívnu koreláciu - čím vyšší príjem, tým vyššia cena domu

Poznámka: Vzdialenosť bodov od priamky predstavuje chybu predikcie modelu

Poznámka: Zhluky bodov ďaleko od priamky môžu indikovať oblasti, kde model nie je presný

Poznámka: Hustota bodov ukazuje distribúciu dát - kde máme najviac pozorovaní

Vyhodnotenie modelu

Výpočet metrik

```
from sklearn.metrics import r2_score,  
mean_squared_error  
  
r2 = r2_score(y_test, y_pred)  
  
mse = mean_squared_error(y_test, y_pred)  
  
print("R2 skóre:", r2)  
  
print("MSE:", mse)
```

Poznámka: Tento kód používa scikit-learn knižnicu pre výpočet dvoch základných metrik. Funkcia `r2_score` porovnáva predikcie s reálnymi hodnotami a `mean_squared_error` počíta priemernú kvadratickú chybu.

Interpretácia R²

R² udáva podiel vysvetlenej variability na celkovej variabilite. Hodnota blízka 1 znamená, že model dobre vysvetľuje variabilitu v dátach. Hodnota blízka 0 znamená, že model nevysvetľuje takmer žiadnu variabilitu.

Poznámka: Pri hodnote R² = 0.75 model vysvetľuje 75% variability v dátach. V praxi sa za dobrý model často považuje taký, ktorý má R² > 0.7, no závisí to od konkrétnej domény a úlohy.

Interpretácia MSE

MSE (Mean Squared Error) je priemerná hodnota štvorcov rozdielov medzi predikovanými a skutočnými hodnotami. Nižšie hodnoty znamenajú lepší model. MSE je v jednotkách závislej premennej na druhú.

Poznámka: MSE je citlivá na odľahlé hodnoty kvôli použitiu druhej mocniny. Pre porovnanie s inými modelmi je často užitočné použiť RMSE (Root Mean Squared Error), čo je odmocnina z MSE, ktorá vracia hodnotu v pôvodných jednotkách závislej premennej.



Interpretácia koeficientov modelu

Správna interpretácia koeficientov je kľúčová pre pochopenie výsledkov regresnej analýzy.

Matematický model

Náš model má tvar:

$$\text{cena_domu} = \text{intercept} + \text{koeficient} \times \text{medián_príjmu}$$

Napríklad, ak je intercept = 0.5 a koeficient = 0.8, potom:

$$\text{cena_domu} = 0.5 + 0.8 \times \text{medián_príjmu}$$

Poznámka: Jednotky koeficientov závisia od jednotiek premenných. Pre lepšiu porovnatelnosť sa niekedy používa štandardizácia premenných.

Poznámka: Koeficienty odrážajú koreláciu, nie nevyhnutne kauzalitu. Vysoký koeficient neznamená automaticky príčinnú súvislosť medzi premennými.

Praktická interpretácia

Koeficient (sklon priamky) vyjadruje, o koľko sa zmení cena domu pri zmene mediánu príjmu o jednotku. Napríklad, koeficient 0.8 znamená, že pri zvýšení mediánu príjmu o 1 jednotku sa cena domu zvýší o 0.8 jednotky.

Intercept (priesečník s osou y) predstavuje predpovedanú cenu domu, keď je medián príjmu nulový. V praxi môže byť táto hodnota mimo rozsahu dát a nemusí mať zmysluplnú interpretáciu.

Poznámka: Pri interpretácii koeficientov je dôležité overiť, či sú štatisticky významné (p-hodnota) a zohľadniť aj ich intervaly spoľahlivosti.

Limity jednoduchej lineárnej regresie

Predpoklad linearity

Model predpokladá lineárny vzťah medzi premennou a cieľovou hodnotou, čo nemusí vždy platiť v reálnych dátach.

Poznámka: Napríklad vzťah medzi vekom a spotrebou liekov môže byť exponenciálny, nie lineárny. V takýchto prípadoch je potrebné použiť transformáciu premenných alebo nelineárne modely.

Obmedzená expresivita

Jednoduchá lineárna regresia môže zachytiť len lineárne vzťahy, nie komplexnejšie vzory v dátach.

Poznámka: Pri analýze ekonomických cyklov, sezónnych výkyvov alebo iných periodických javov je jednoduchá lineárna regresia výrazne limitujúca. Polynomiálne funkcie alebo gausovské procesy môžu byť vhodnejšou alternatívou.

Citlivosť na odľahlé hodnoty

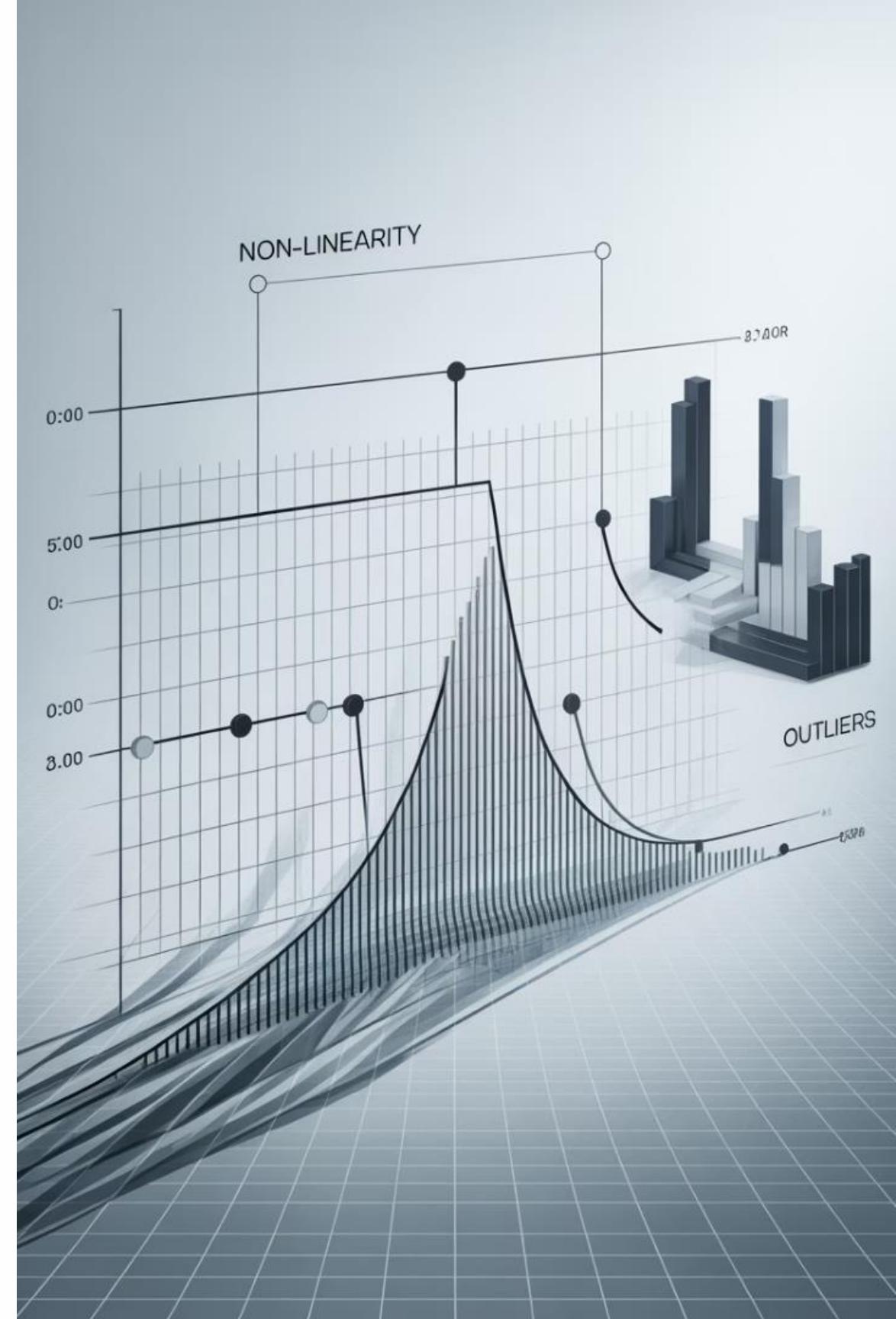
Extrémne hodnoty môžu výrazne ovplyvniť sklon regresnej priamky a skresliť predikcie.

Poznámka: Jediná extrémna hodnota môže zmeniť koeficient sklonu až o desiatky percent. Robustné metódy ako RANSAC alebo kvantilová regresia sú menej citlivé na outlierov.

Ignorovanie ďalších premenných

Model berie do úvahy len jednu premennú, čo môže viesť k nedostatočnému vysvetleniu variability v dátach.

Poznámka: Pri predikcii cien nehnuteľností napríklad nestačí zohľadniť len rozlohu, ale aj lokalitu, vek budovy, počet izieb a ďalšie faktory. Ignorovanie týchto premenných viedie k problému vynechaných premenných (omitted variable bias).





Čo sa naučíme?

1. Ako začať používať lineárnu regresiu?
2. Z akých častí sa skladá lineárny regresný model?
3. Ako sa model trénuje (fituje) v scikit-learn?
4. Ako interpretovať výstupy modelu (koeficienty a intercept)?
5. Ako sa vyhodnocuje presnosť lineárneho modelu?
6. Aké sú výhody a nevýhody lineárnej regresie?
7. Profit

Ako používať Viacnásobnú Regresiu

AKREDITOVANÝ KURZ

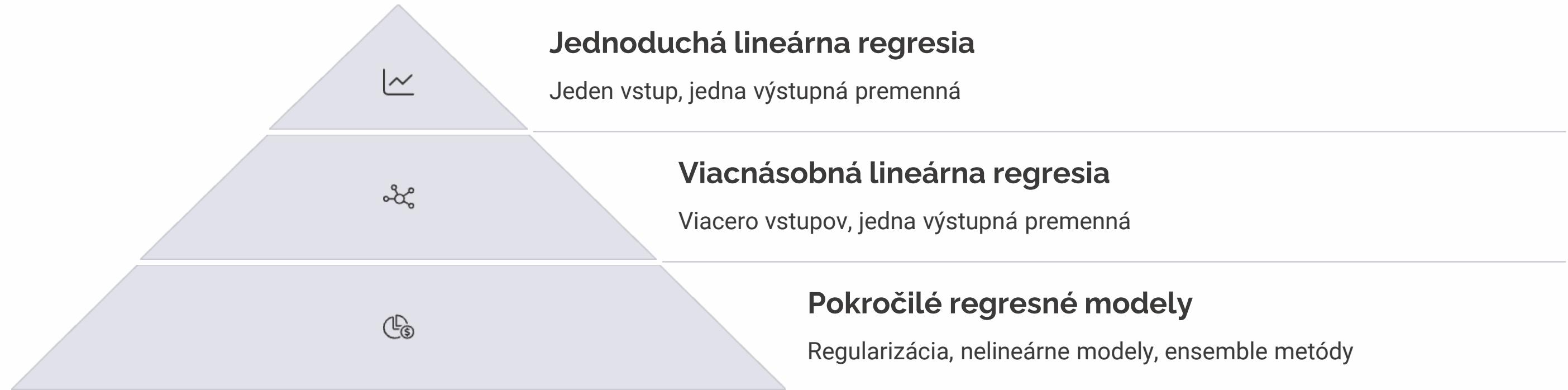




Čo sa naučíme?

1. Čo je viacnásobná regresia a kedy sa používa?
2. Ako sa líši viacnásobná regresia od jednoduchej regresie?
3. Ako sa vyberajú vhodné vstupné premenné do modelu?
4. Ako škálovať vstupné premenné a prečo je to dôležité?
5. Ako sa vyhodnocuje výkon modelu s viacerými premennými?
6. Na čo si dať pozor pri viacnásobnej regresii?
7. Profit

Viacnásobná regresia

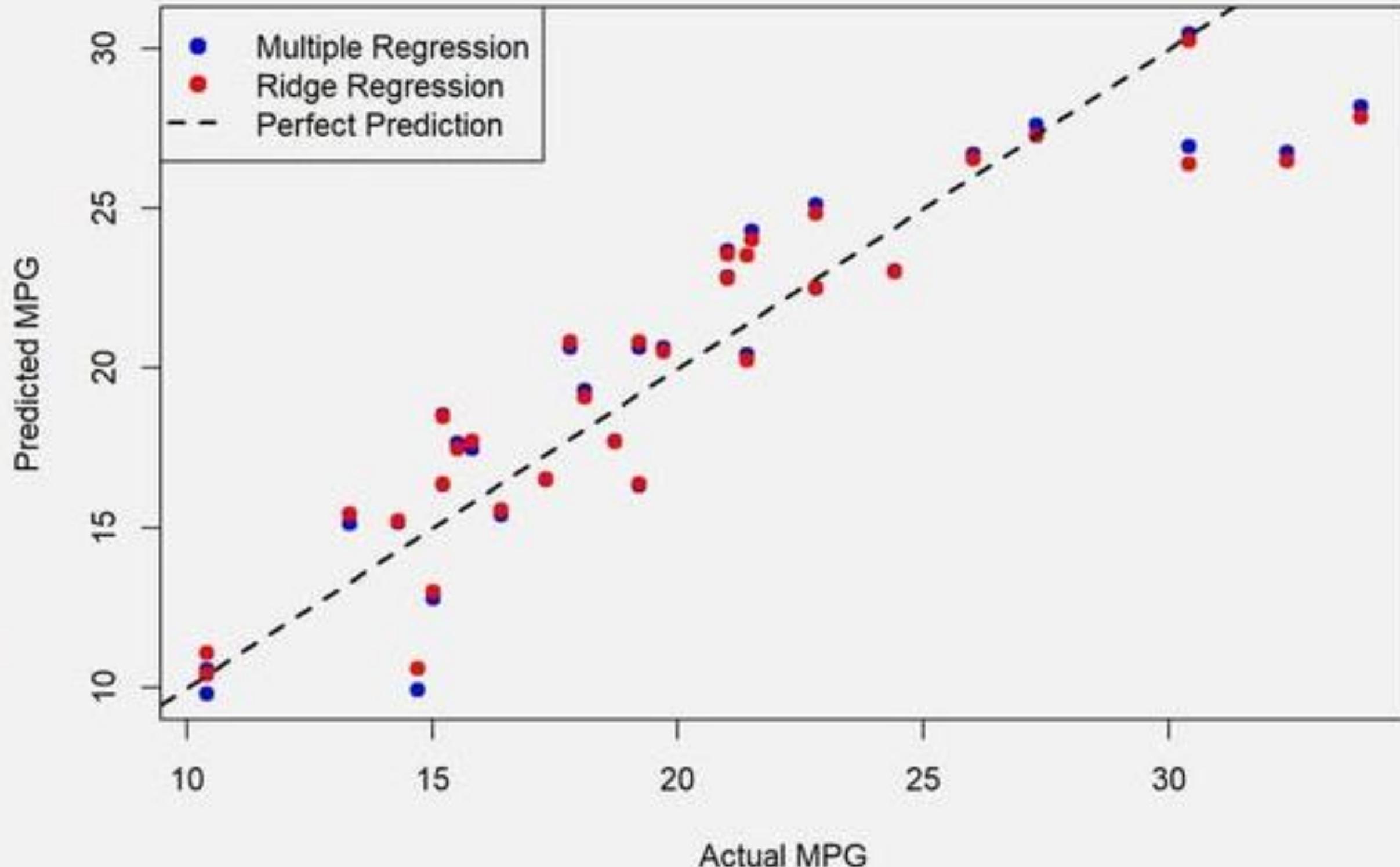


Poznámky k jednoduchej lineárnej regresii: Matematický vzorec $y = \beta_0 + \beta_1 x + \varepsilon$, kde β_0 je intercept, β_1 je sklon priamky a ε je chyba. Typické použitie zahŕňa predikciu cien nehnuteľností na základe rozlohy alebo predikciu spotreby paliva na základe hmotnosti vozidla.

Poznámky k viacnásobnej lineárnej regresii: Rozširuje jednoduchý model pomocou viacerých prediktorov vo forme $y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \dots + \beta_n x_n + \varepsilon$. Pomáha lepšie vysvetliť variabilitu v dátach a poskytuje presnejšie predikcie. Príklad: predikcia ceny domu na základe rozlohy, počtu izieb a vzdialenosť od centra.

Poznámky k pokročilým regresným modelom: Riešia problémy ako multikolinearita, overfitting a komplexné nelineárne vzťahy. Regularizačné techniky (Ridge, Lasso, ElasticNet) pridávajú penalizáciu pre komplexné modely. Nelineárne modely zachytávajú zložitejšie vzťahy v dátach. Ensemble metódy kombinujú výsledky viacerých modelov pre vyššiu presnosť.

Actual vs Predicted MPG



Matematický model viacnásobnej regresie

Vzorec viacnásobnej regresie

Viacnásobná regresia rozširuje jednoduchú lineárnu regresiu o ďalšie premenné:

$$y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \beta_3 x_3 + \dots + \varepsilon$$

kde β_0 je intercept, $\beta_1, \beta_2, \beta_3, \dots$ sú koeficienty pre jednotlivé premenné a ε je chyba predikcie.

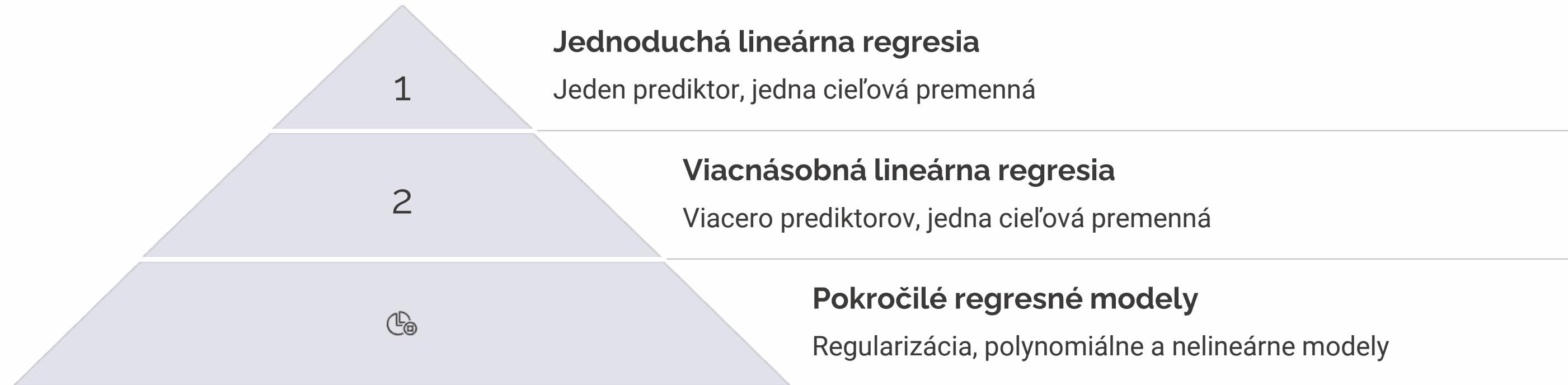
Príklad: Predikcia ceny nehnuteľnosti

$$\text{Cena} = \beta_0 + \beta_1 \times \text{rozloha} + \beta_2 \times \text{vzdialenosť_od_centra} + \beta_3 \times \text{počet_izieb} + \varepsilon$$

Každý koeficient vyjadruje, ako sa zmení cena pri zmene danej premennej o jednotku (pri zachovaní ostatných premenných).

Rozšírenie na viacnásobnú regresiu

Regresné modely sa postupne vyvíjajú od jednoduchých k zložitejším, pričom každý typ rieši špecifické analytické potreby.



Poznámky:

- **Jednoduchá lineárna regresia:** Základný model tvaru $y = \beta_0 + \beta_1 x + \varepsilon$, ktorý popisuje vzťah medzi dvomi premennými. Vhodný pre úvodné analýzy a jednoduché predikcie.
- **Viacnásobná lineárna regresia:** Rozšírenie jednoduchej regresie na viacero prediktorov: $y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \dots + \beta_n x_n + \varepsilon$. Zachytáva komplexnejšie vzťahy a interakcie medzi premennými.
- **Pokročilé regresné modely:** Riešia obmedzenia lineárnych modelov pomocou regularizácie (Ridge, Lasso), polynomiálnych rozšírení a nelineárnych techník, ktoré lepšie modelujú zložité dátové vzory.

Použitie všetkých premenných z datasetu

Príprava dát

```
X = df[data.feature_names] # Všetky dostupné premenné
```

```
y = df["target"]
```

Poznámka: V tomto kroku extrahujeme všetky prediktorové premenné z datasetu a definujeme našu cieľovú premennú (ceny domov). Použitie všetkých dostupných premenných nám umožní vytvoriť komplexnejší model.

Rozdelenie dát

```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
```

Poznámka: Dáta rozdeľujeme v pomere 80:20 na trénovaciu a testovaciu množinu. Parameter random_state=42 zabezpečuje reproducibilnosť výsledkov pri opakovanom spustení kódu.

Škálovanie premenných

```
scaler = StandardScaler()
```

```
X_train_scaled = scaler.fit_transform(X_train)
```

```
X_test_scaled = scaler.transform(X_test)
```

Poznámka: Škálovanie je kľúčové pre viacnásobnú regresiu, pretože premenné majú rôzne rozsahy. StandardScaler štandardizuje hodnoty tak, aby mali priemer 0 a štandardnú odchýlku 1, čo zlepšuje konvergenciu modelu.

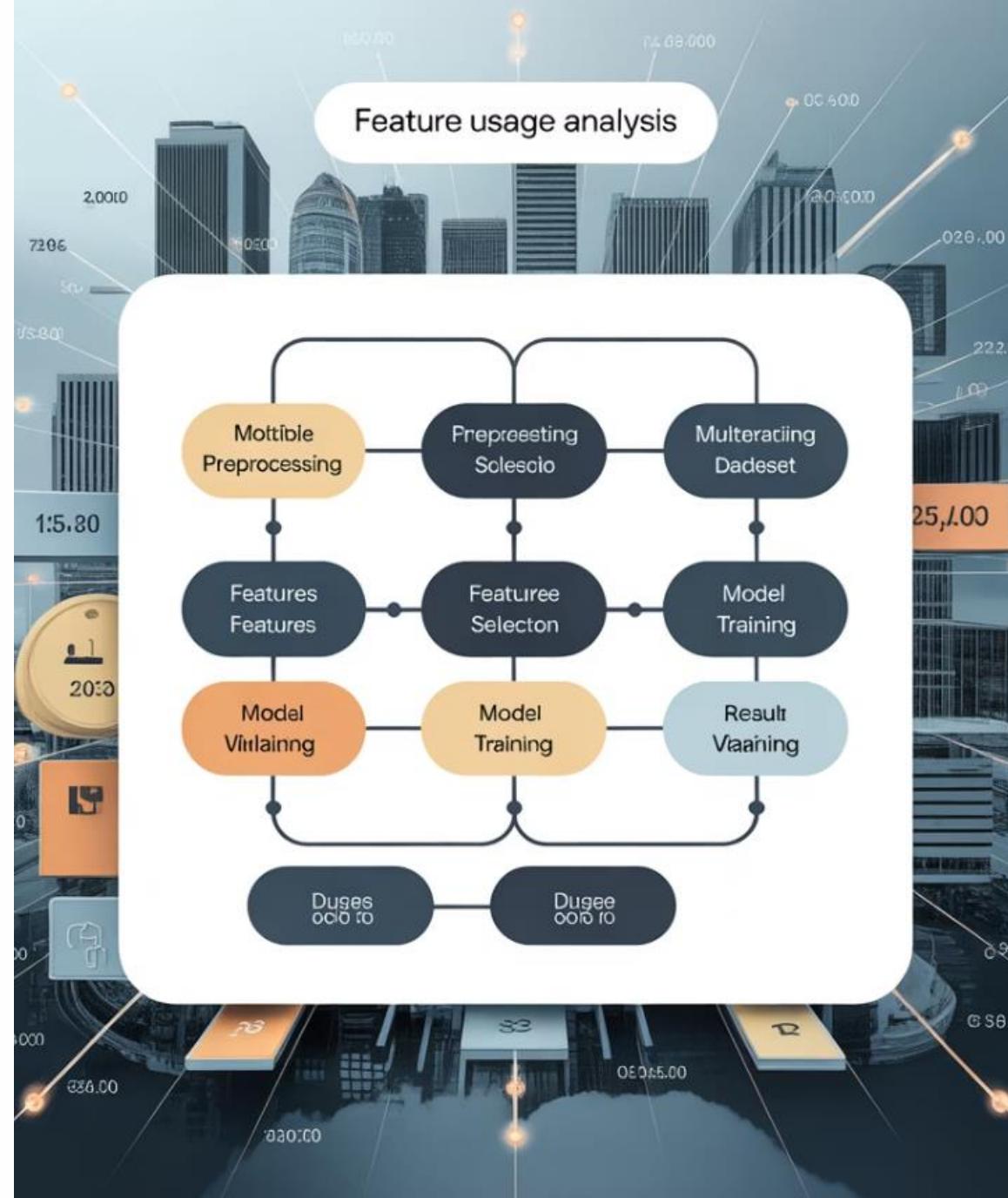
Tréning modelu

```
model = LinearRegression()
```

```
model.fit(X_train_scaled, y_train)
```

Poznámka: Na škálovaných dátach trénujeme model lineárnej regresie, ktorý bude hľadať optimálne koeficienty pre všetky premenné súčasne. Tento viacozmerný prístup dokáže zachytiť komplexnejšie vzťahy medzi premennými a cieľovou hodnotou.

California Housing Dataset Regression



Porovnanie jednoduchej a viacnásobnej regresie

Jednoduchá regresia (len MedInc)

```
model_simple = LinearRegression()  
  
model_simple.fit(X_train[["MedInc"]], y_train)  
  
y_pred_simple = model_simple.predict(X_test[["MedInc"]])  
  
r2_simple = r2_score(y_test, y_pred_simple)  
  
print("R2 (jednoduchá):", r2_simple)
```

Poznámka: Jednoduchá regresia používa len jednu premennú (mediánový príjem) na predikciu cieľovej hodnoty. R² hodnota nám ukáže, kol'ko percent variability v dátach dokáže tento jednoduchý model vysvetliť.

Poznámka: Porovnaním R² hodnôt môžeme kvantitatívne určiť, o kol'ko sa zlepšila presnosť modelu pridaním ďalších premenných. Táto metrika predstavuje podiel vysvetlenej variability - čím bližšie k 1, tým lepší model.

Viacnásobná regresia (všetky premenné)

```
model_multiple = LinearRegression()  
  
model_multiple.fit(X_train_scaled, y_train)  
  
y_pred_multiple = model_multiple.predict(X_test_scaled)  
  
r2_multiple = r2_score(y_test, y_pred_multiple)  
  
print("R2 (viacnásobná):", r2_multiple)
```

Poznámka: Viacnásobná regresia využíva všetky dostupné premenné, čo zvyčajne vedie k presnejším predikciám. Škálované dáta (X_train_scaled) zabezpečujú, že všetky premenné majú rovnakú váhu. Očakávame vyššiu hodnotu R² ako pri jednoduchej regresii.

Analýza koeficientov viacnásobnej regresie

Získanie koeficientov

```
# Vytvoríme DataFrame pre prehľadné zobrazenie koeficientov  
  
coef_df = pd.DataFrame({  
  
    'Feature': X.columns, # Názvy premenných  
  
    'Coefficient': model.coef_ # Hodnoty koeficientov z modelu  
  
})  
  
# Zoradíme koeficienty od najväčšieho po najmenší  
  
coef_df = coef_df.sort_values('Coefficient', ascending=False)  
  
print(coef_df) # Výpis zoradených koeficientov
```

Vizualizácia koeficientov

```
# Vytvoríme horizontálny stĺpcový graf pre vizuálne porovnanie  
  
plt.figure(figsize=(10, 6)) # Nastavenie veľkosti grafu  
  
plt.barh(coef_df['Feature'], coef_df['Coefficient']) # Horizontálny graf  
  
plt.xlabel('Koeficient') # Popis osi x  
  
plt.ylabel('Premenná') # Popis osi y  
  
plt.title('Koeficienty viacnásobnej regresie') # Názov grafu  
  
plt.grid(axis='x') # Mriežka pre lepšiu čitateľnosť  
  
plt.show() # Zobrazenie grafu  
  
# Veľkosť a znamienko koeficientov ukazujú silu a smer vplyvu  
premenných
```

Poznámka: Kladné koeficienty znamenajú pozitívny vplyv premennej na cieľovú hodnotu, zatiaľ čo záporné koeficienty indikujú negatívny vplyv. Absolútна hodnota koeficientu určuje silu tohto vplyvu.

Implementácia viacnásobnej regresie



Príprava dát

Načítanie a spracovanie dát s viacerými vstupnými premennými.

Dôležité je identifikovať chýbajúce hodnoty a správne ich nahradíť. Používame metódy ako `pandas.read_csv()` a `DataFrame.fillna()` alebo `SimpleImputer` z `scikit-learn`.

Škálovanie premenných

Normalizácia vstupných premenných pre lepšiu konvergenciu modelu.

Pomocou `StandardScaler` alebo `MinMaxScaler` transformujeme premenné na rovnakú škálu, čo zlepšuje numerickú stabilitu a rýchlosť konvergencie algoritmu.

Trénovanie modelu

Použitie rovnakého API ako pri jednoduchej regresii: `model.fit(X_train, y_train)`.

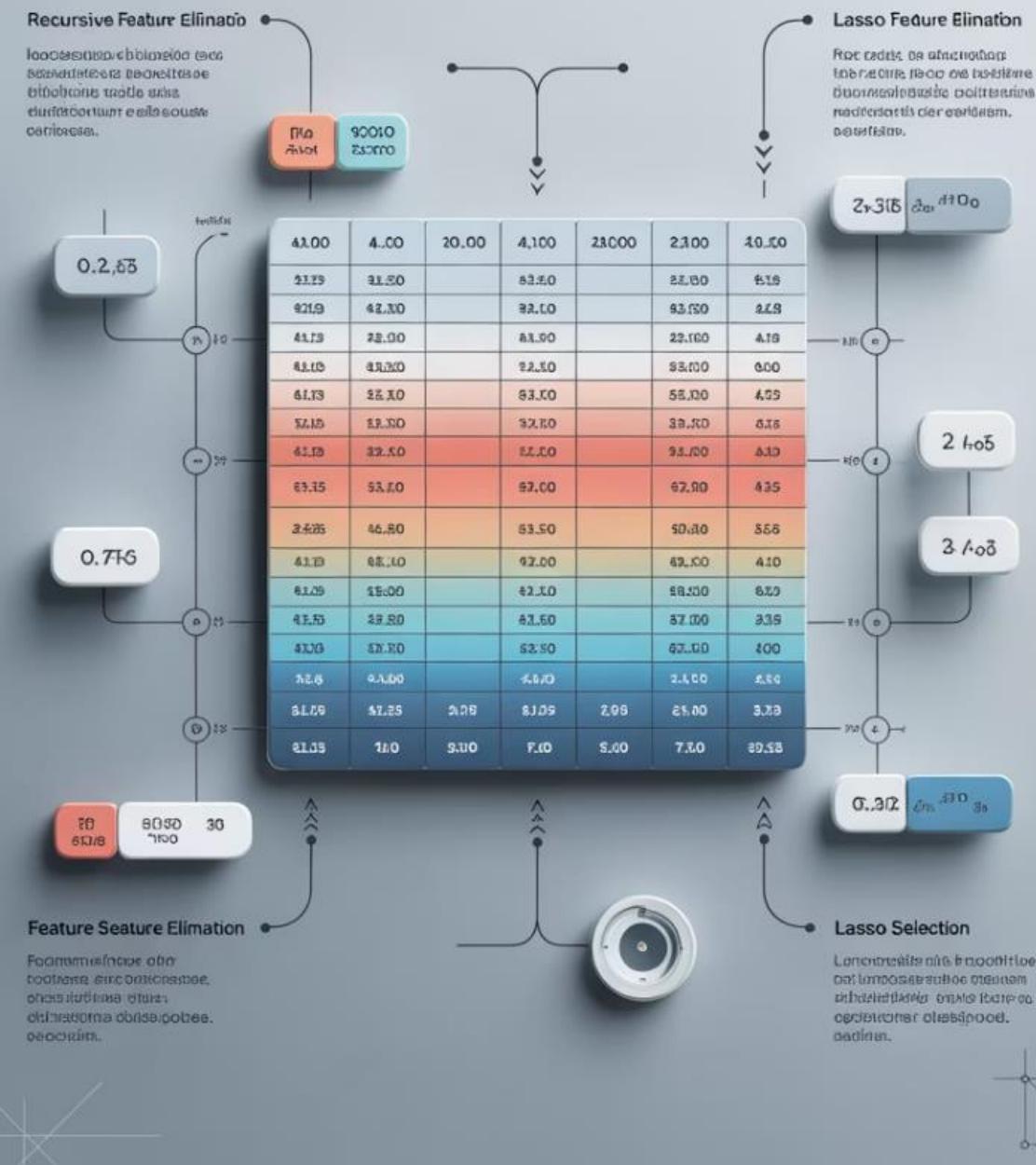
V `scikit-learn` používame `LinearRegression()`, kde `X_train` je matica s viacerými stĺpcami. Pre lepšiu generalizáciu môžeme využiť aj cross-validáciu pomocou `cross_val_score()`.

Interpretácia koeficientov

Analýza koeficientov pre pochopenie vplyvu jednotlivých premenných.

Koeficienty získame cez `model.coef_`, pričom ich hodnota vyjadruje vplyv danej premennej na výsledok. Štatistickú významnosť môžeme posúdiť pomocou p-hodnôt z `statsmodels.api.OLS`.

Feature Selection Methods



Výber dôležitých vstupných premenných

Korelácie

Analýza korelácií medzi premennými pomocou korelačnej matice a heatmaps: `sns.heatmap(df.corr())`. Vysoké absolútne hodnoty korelácií môžu indikovať multikolinearitu, čo môže negatívne ovplyvniť výkon modelu.

SelectKBest

Výber k najlepších premenných na základe štatistických testov. Používa metriky ako chi-kvadrát alebo ANOVA F-hodnoty na určenie najinformatívnejších premenných pre predikciu cieľovej premennej.

Recursive Feature Elimination (RFE)

Rekurzívne odstraňovanie najmenej dôležitých premenných. Táto metóda trénuje model, odstraňuje najslabšie premenné a opakuje proces, kým nedosiahne požadovaný počet premenných, čím zvyšuje presnosť a efektivitu modelu.

Lasso výber

Využitie L1 regularizácie na automatický výber premenných. Lasso metóda "penalizuje" koeficienty modelu, čím tlačí hodnoty menej významných premenných k nule a efektívne vykonáva výber premenných počas trénovalia modelu.

Interpretácia koeficientov viacnásobnej regresie

Význam koeficientov

Koeficient pre premennú X_i vyjadruje, o kol'ko sa zmení cieľová premenná pri zmene X_i o jednotku, za predpokladu, že všetky ostatné premenné zostanú konštantné.

Poznámka: Napríklad, ak je koeficient pre premennú "vzdelanie" rovný 0.5, potom zvýšenie vzdelania o 1 rok viedie k zvýšeniu predpovedanej mzdy o 0.5 jednotiek, pri zachovaní ostatných premenných.

Štandardizované koeficienty

Ked' sú premenné štandardizované, koeficienty sú porovnateľné a indikujú relatívnu dôležitosť premenných. Väčšia absolútna hodnota koeficientu znamená väčší vplyv na cieľovú premennú.

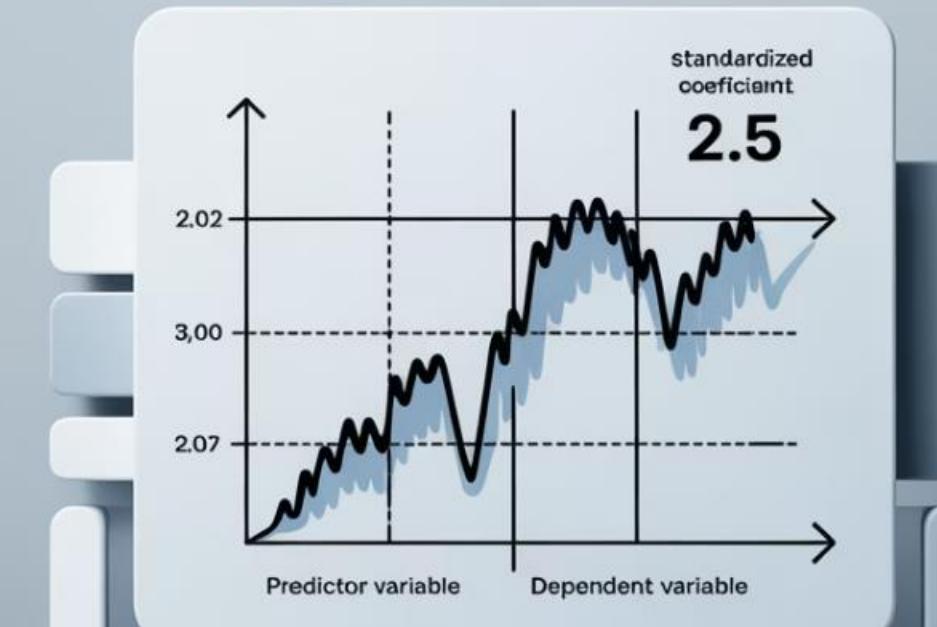
Poznámka: Štandardizácia sa vykonáva odčítaním priemeru a delením štandardou odchýlkou (z-skóre). Umožňuje porovnávať vplyv premenných s rôznymi mernými jednotkami.

Pozitívne vs. negatívne koeficienty

Pozitívny koeficient znamená, že zvýšenie hodnoty premennej viedie k zvýšeniu cieľovej premennej. Negatívny koeficient znamená, že zvýšenie hodnoty premennej viedie k zníženiu cieľovej premennej.

Poznámka: Príklad - pozitívny koeficient pre "prax" znamená, že s rastúcimi rokmi praxe sa zvyšuje predpovedaný príjem. Negatívny koeficient pre "vek zariadenia" znamená, že staršie zariadenia sú spojené s nižšou produktivitou.

Interpretation of Standardized Coefficients



P-Value

Standardized coefficient očiňte významnosť jednotlivých premenných v modeli. Vysoké hodnoty sú obvykle súvisiace s významnosťou.

P-Value

Standardized coefficient očiňte významnosť jednotlivých premenných v modeli. Vysoké hodnoty sú obvykle súvisiace s významnosťou.

R-Squared Value

Standardized coefficient očiňte významnosť jednotlivých premenných v modeli. Vysoké hodnoty sú obvykle súvisiace s významnosťou.

Korelačná matica a heatmapa

Čo je korelačná matica?

Korelačná matica zobrazuje Pearsonove korelačné koeficienty medzi všetkými párami premenných v datasete. Koeficienty sa pohybujú od -1 (dokonalá negatívna korelácia) cez 0 (žiadna korelácia) po 1 (dokonalá pozitívna korelácia).

Vytvorenie heatmapy v seaborn

Heatmapa je vizuálna reprezentácia korelačnej matice, kde farby reprezentujú hodnoty korelácií:

```
import seaborn as sns
```

```
import matplotlib.pyplot as plt
```

```
plt.figure(figsize=(10, 8))
```

```
sns.heatmap(df.corr(), annot=True, cmap='coolwarm')
```

```
plt.title('Korelačná matica')
```

```
plt.show()
```

Metódy výberu premenných

SelectKBest

Výber k najlepších premenných na základe štatistických testov:

```
from sklearn.feature_selection import SelectKBest, f_regression  
selector = SelectKBest(f_regression, k=5)  
  
X_new = selector.fit_transform(X, y)  
  
selected_features = X.columns[selector.get_support()]
```

Poznámka: Táto metóda hodnotí vzťah medzi každou premennou a cieľovou premennou nezávisle. Parameter k určuje počet najlepších premenných, ktoré majú byť vybrané. Vhodná pre rýchlu a jednoduchú selekciu premenných.

Poznámka: Pri výbere metódy je dôležité zvážiť veľkosť datasetu, výpočtové nároky a charakter problému. Kombinácia viacerých metód často viedie k robustnejším výsledkom.

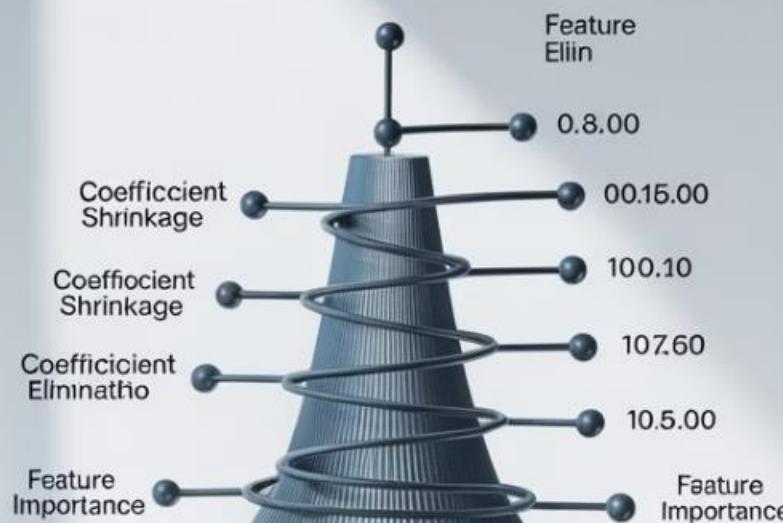
Recursive Feature Elimination (RFE)

Rekurzívne odstraňovanie najmenej dôležitých premenných:

```
from sklearn.feature_selection import RFE  
from sklearn.linear_model import LinearRegression  
  
model = LinearRegression()  
  
selector = RFE(model, n_features_to_select=5)  
  
selector.fit(X, y)  
  
selected_features = X.columns[selector.support_]
```

Poznámka: RFE pracuje rekurzívne - postupne odstraňuje najmenej dôležité premenné na základe váh modelu. Výhodou je, že berie do úvahy interakcie medzi premennými, ale je výpočtovo náročnejšia ako SelectKBest.

Lasso regression



Lasso výber premenných

Príprava dát

Škálovanie vstupných premenných pre lepšie výsledky.

Poznámka: Štandardizácia premenných (odčítanie priemeru a delenie štandardou odchýlkou) je kľúčová pre Lasso, keďže metóda je citlivá na škálu vstupných premenných. Neštandardizované dáta môžu viest k skresleným výsledkom.

Trénovanie Lasso modelu

Použitie Lasso regresie s vhodným parametrom alpha.

Poznámka: Parameter alpha kontroluje silu penalizácie. Vyššie hodnoty alpha vedú k väčšiemu "zriedeniu" modelu (viac koeficientov sa stane nulami). Optimálnu hodnotu alpha možno nájsť pomocou cross-validácie (napr. LassoCV).

Analýza koeficientov

Premenné s nulovými koeficientmi sú eliminované z modelu.

Poznámka: Kľúčovou vlastnosťou Lasso je schopnosť úplne eliminovať irrelevantné premenné nastavením ich koeficientov presne na nulu. Táto vlastnosť robí Lasso užitočným nielen pre predikciu, ale aj pre interpretáciu a pochopenie dát.

Výber premenných

Použitie premenných s nenulovými koeficientmi pre finálny model.

Poznámka: Po aplikácii Lasso možno vybrať premenné s nenulovými koeficientmi a použiť ich v inom, potenciálne interpretovateľnejšom modeli. Tento prístup kombinuje výhody automatického výberu premenných s flexibilitou voľby finálneho modelu.

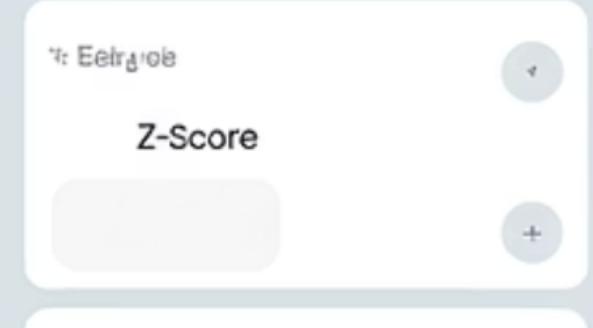
Zníženie rozmernosti pomocou PCA

Čo je PCA?

Principal Component Analysis (PCA) je technika na zníženie rozmernosti dát. Transformuje pôvodné premenné na nové, nekorelované premenné (hlavné komponenty), ktoré zachytávajú maximálne množstvo variability v dátach.

Implementácia v scikit-learn

```
from sklearn.decomposition import PCA  
  
pca = PCA(n_components=2)  
  
X_pca = pca.fit_transform(X_scaled)  
  
print("Vysvetlená variancia:", pca.explained_variance_ratio_)  
  
print("Kumulatívna vysvetlená variancia:",  
np.sum(pca.explained_variance_ratio_))
```



Normalizácia dát pre regresiu

Normalizácia je kľúčový krok predspracovania dát, ktorý výrazne zlepšuje výkon a interpretovateľnosť regresných modelov.

Prečo normalizovať dátá?

Vstupné atribúty by mali mať podobný rozsah, aby:

- Algoritmy konvergovali rýchlejšie
- Premenné s väčším rozsahom nedominovali
- Regularizácia fungovala správne
- Koeficienty boli porovnatelné

Poznámka: Bez normalizácie môžu atribúty s veľkými hodnotami (napr. príjem v eurách) prevážiť atribúty s malými hodnotami (napr. vek), čo skresluje výsledky modelu.

StandardScaler

Transformuje dátá tak, aby mali nulovú strednú hodnotu a jednotkovú smerodajnú odchýlku:

```
from sklearn.preprocessing import StandardScaler
scaler = StandardScaler()
X_train_scaled = scaler.fit_transform(X_train)
X_test_scaled = scaler.transform(X_test)
```

Poznámka: Vhodný pre algoritmy, ktoré predpokladajú normálne rozdelenie dát, ako napr. lineárna regresia a SVM. Je odolný voči odľahlým hodnotám (outlierom).

MinMaxScaler

Transformuje dátá do rozsahu [0, 1]:

```
from sklearn.preprocessing import MinMaxScaler
scaler = MinMaxScaler()
X_train_scaled = scaler.fit_transform(X_train)
X_test_scaled = scaler.transform(X_test)
```

Poznámka: Vhodný pre algoritmy, ktoré vyžadujú nezáporné hodnoty, napr. niektoré typy neurónových sietí. Je citlivý na odľahlé hodnoty, ale zachováva pôvodné rozdelenie.

Dôležité: Normalizáciu je potrebné vykonať iba na trénovacích dátach a potom aplikovať rovnakú transformáciu na testovacie dátá, aby sa zabránilo úniku informácií.

Dôležitosť normalizácie pri regularizácii

Bez normalizácie

Premenné s väčším rozsahom budú menej penalizované regularizáciou, čo viedie k suboptimálnym výsledkom.

Koeficienty nie sú priamo porovnateľné a model môže byť nestabilný.

Poznámka: Keď používame napr. L1 alebo L2 regularizáciu, veľké atribúty budú dominovať, pretože ich zmena má väčší vplyv na cieľovú funkciu. Toto môže viesť k preučeniu (overfitting) a zlej interpretateľnosti modelu.

Dôležitá poznámka: Pri použití normalizácie je potrebné pamätať na aplikovanie rovnakej transformácie na trénovacie aj testovacie dát. Vždy používajte `fit_transform()` na trénovacích dátach a len `transform()` na testovacích dátach.

S normalizáciou

Všetky premenné sú na rovnakej škále, takže regularizácia ich penalizuje rovnomerne. Koeficienty sú priamo porovnateľné a model je stabilnejší a lepšie generalizuje na nové dátá.

Poznámka: Normalizácia (napr. StandardScaler alebo MinMaxScaler) zabezpečuje, že algoritmus bude konvergovať rýchlejšie a hyperparametre regularizácie budú mať konzistentný účinok na všetky premenné bez ohľadu na ich pôvodný rozsah.

Supervised Learning: Základné pojmy

Regresia

Ciel: Predikovať spojité číselné hodnoty.

Použitý model: LinearRegression

Vizualizácia: Skutočné hodnoty (modré body) a predikovaná priamka (červená)

Poznámka: Pri regresii hľadáme funkciu, ktorá najlepšie vystihuje vzťah medzi vstupnými premennými (X) a výstupnou premennou (y). Kvalita modelu sa hodnotí metrikami ako MSE, MAE alebo R^2 .

Klasifikácia

Ciel: Predikovať kategóriu (0 alebo 1).

Použitý model: LogisticRegression

Vizualizácia: Body sú farebne odlišené podľa predikcie (farba) a skutočného výsledku (tvar)

Poznámka: Klasifikačné modely vytvárajú hranice rozhodnutia v priestore príznakov. Hodnotíme ich pomocou presnosti, recall, precision a F1-skóre. Pre viac triednu klasifikáciu možno použiť stratégie ako one-vs-rest alebo one-vs-one.

Hodnotenie modelov

R^2

Koeficient determinácie

Udáva, ako dobre model vysvetľuje variabilitu dát. Hodnoty sa pohybujú od 0 do 1, pričom vyššie hodnoty znamenajú lepší model.

MSE

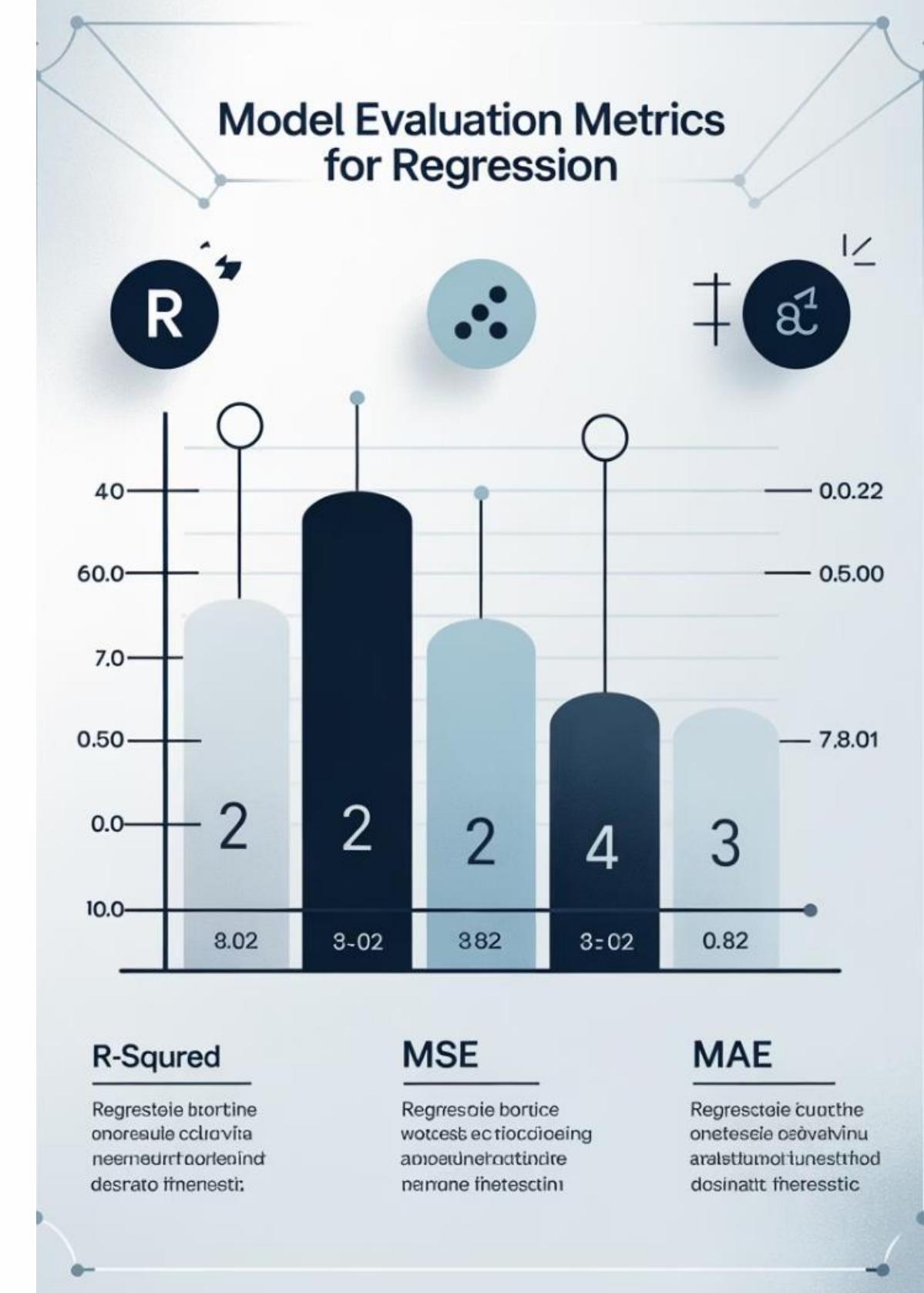
Priemerná štvorcová chyba

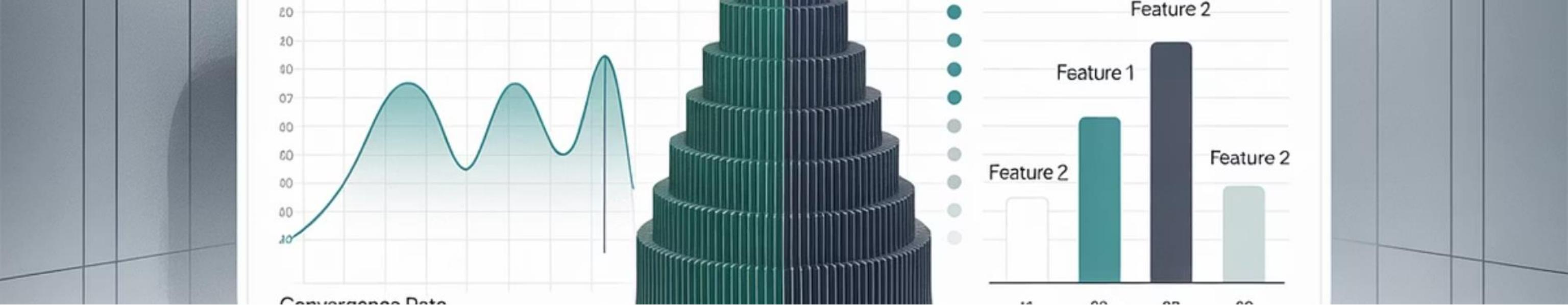
Priemerná hodnota štvorcov rozdielov medzi predikovanými a skutočnými hodnotami. Nižšie hodnoty znamenajú lepší model.

MAE

Priemerná absolútна chyba

Priemerná hodnota absolútnych rozdielov medzi predikovanými a skutočnými hodnotami. Nižšie hodnoty znamenajú lepší model.





Na čo slúži normalizácia vstupov



Odstránenie mierky

Premenné v datasete môžu mať rôzne jednotky a rozsahy (napr. počet izieb vs. príjem). Normalizácia zabezpečuje porovnateľnosť premenných.



Zlepšenie konvergencie

Algoritmy ako gradient descent konvergujú rýchlejšie a stabilnejšie, keď sú dátá normalizované.



Zabránenie dominancii

Premenná s veľkým rozsahom by mohla potlačiť vplyv iných premenných, čo vede k zlým koeficientom a slabému výkonu.

StandardScaler - matematický princíp

Vzorec štandardizácie

StandardScaler transformuje každú premennú tak, aby mala nulovú strednú hodnotu a jednotkovú smerodajnú odchýlku:

$$z = (x - \mu) / \sigma$$

kde μ je priemer a σ je smerodajná odchýlka.

Poznámka: Táto transformácia sa nazýva aj Z-score normalizácia a je jednou z najpoužívanejších metód pre predspracovanie dát v strojovom učení.

Poznámka: Výsledné hodnoty budú väčšinou v rozsahu -3 až +3, čo je výhodné pre mnohé algoritmy.

Príklad

Majme premennú s hodnotami [1, 2, 3, 4, 5].

Priemer $\mu = 3$, smerodajná odchýlka $\sigma = 1.41$.

Po štandardizácii dostaneme:

$$[(1-3)/1.41, (2-3)/1.41, (3-3)/1.41, (4-3)/1.41, (5-3)/1.41]$$

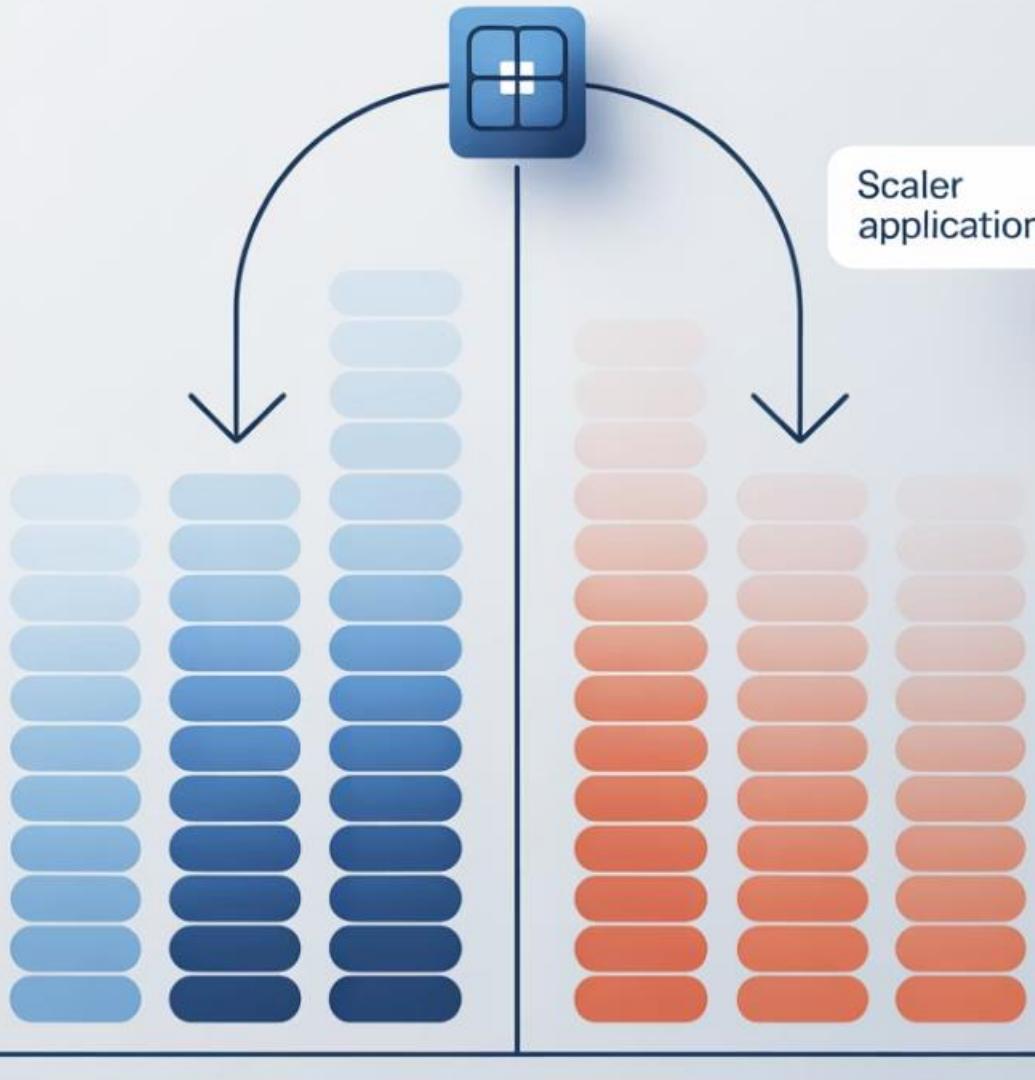
$$= [-1.41, -0.71, 0, 0.71, 1.41]$$

Poznámka: Všimnite si, že pôvodná stredná hodnota 3 sa transformovala na 0.

Poznámka: Táto transformácia zachováva tvar distribúcie a relatívne vzdialenosť medzi hodnotami.

Poznámka: V praxi sa implementuje pomocou sklearn.preprocessing.StandardScaler v Pythone.

Test data Normalization process



Training data

Test data

Poznámka k normalizácii testovacích dát

Trénovacia množina

Použijeme `fit_transform()` na výpočet štatistik a transformáciu dát.

Táto metóda vypočíta priemer a smerodajnú odchýlku trénovacích dát a následne ich použije na štandardizáciu tých istých dát. Ide o kľúčový krok pre správne nastavenie skalera.

Uloženie štatistik

Scaler si zapamäta priemery a smerodajné odchýlky z trénovacej množiny.

Tieto hodnoty sú uložené ako atribúty objektu (napr. `scaler.mean_`, `scaler.scale_`) a budú použité pre všetky budúce transformácie. Je dôležité používať rovnaké transformačné parametre pre trénovaciu aj testovaciu množinu.

Testovacia množina

Použijeme len `transform()` s už naučenými štatistikami.

Toto zabezpečí, že testovacie dáta budú transformované rovnakým spôsobom ako trénovacie dátá. Tým zabezpečíme konzistentnú reprezentáciu vo feature space a správne fungovanie modelu.

Dôležité upozornenie

Nikdy nepoužívame `fit()` na testovacej množine, aby nedošlo k úniku informácií!

Použitie `fit()` alebo `fit_transform()` na testovacej množine by viedlo k úniku informácií (data leakage), pretože model by získal informácie o distribúcii testovacích dát, čo by spôsobilo nadhodnotenie výkonnosti modelu a neobjektívne výsledky.

Správna sekvencia príkazov pre normalizáciu je teda: `scaler.fit_transform(X_train)` pre trénovacie dátá a následne `scaler.transform(X_test)` pre testovacie dátá, nikdy nie `scaler.fit_transform(X_test)`.

Kedy normalizácia nie je potrebná

Rozhodovacie stromy

Modely založené na stromoch (DecisionTree, RandomForest) nie sú citlivé na škálu premenných, pretože pracujú s rozdelením dát na základe prahových hodnôt, nie s absolútnymi hodnotami.

Poznámka: Pri experimentoch s RandomForest môžete použiť rovnaké dátá bez normalizácie a dosiahnuť rovnaké alebo podobné výsledky ako s normalizovanými dátami.

Naive Bayes

Niekteré implementácie Naive Bayes klasifikátorov nie sú citlivé na škálu premenných, pretože pracujú s pravdepodobnosťami, nie s absolútными hodnotami.

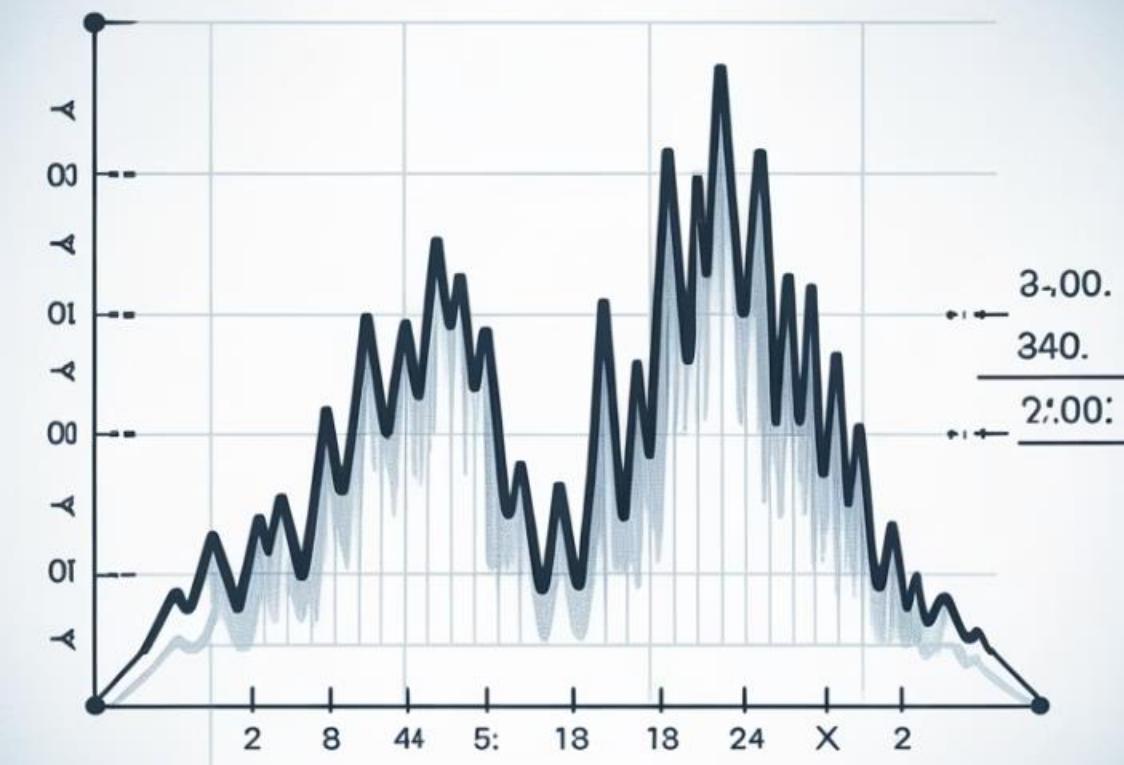
Poznámka: Gaussian Naive Bayes môže byť mierne ovplyvnený škálovaním, zatiaľ čo Multinomial a Bernoulli Naive Bayes sú väčšinou invariantné voči škálovaniu.

Algoritmy založené na vzdialostiach

Naopak, algoritmy ako K-nearest neighbors, K-means, SVM a neurónové siete sú veľmi citlivé na škálu premenných a vyžadujú normalizáciu.

Poznámka: Bez normalizácie môžu atribúty s väčšími hodnotami domenovať výpočty vzdialenosťí a spôsobiť nesprávne výsledky alebo pomalú konvergenciu.

SCALE-SENSITIVE ALGORITHMS



SCALE-INVARIANT ALGORITHMS



Tréning regresného modelu s normalizovanými dátami

Import knižníč

```

from sklearn.linear_model import
LinearRegression

from sklearn.preprocessing import
StandardScaler

from sklearn.metrics import
mean_squared_error, r2_score

```

Poznámka: LinearRegression slúži na vytvorenie modelu, StandardScaler na normalizáciu dát a metrics na vyhodnotenie presnosti modelu.

Škálovanie dát

```

scaler = StandardScaler()

X_train_scaled =
scaler.fit_transform(X_train)

```

```
X_test_scaled = scaler.transform(X_test)
```

Poznámka: Dôležité je použiť fit_transform len na trénovacích dátach a transform na testovacích, aby sa zabránilo úniku informácií z testovacích dát.

Tréning modelu

```

model = LinearRegression()

model.fit(X_train_scaled, y_train)

```

Poznámka: Model trénujeme na normalizovaných dátach, čo zlepšuje konvergenciu a stabilitu, najmä pri rozdielnych škálach premenných.

Vyhodnotenie výkonu

```

y_pred = model.predict(X_test_scaled)

print("R² skóre:", r2_score(y_test, y_pred))

print("MSE:",
mean_squared_error(y_test, y_pred))

Poznámka: R² blížiace sa k 1 znamená lepší model. MSE (stredná kvadratická chyba) by mala byť čo najnižšia pre presný model.

```

Vizualizácia predikcií vs. skutočných hodnôt

Kód pre vizualizáciu

```
import matplotlib.pyplot as plt  
  
plt.figure(figsize=(8, 6)) # Nastavenie veľkosti grafu  
  
plt.scatter(y_test, y_pred, alpha=0.3, color='blue') # Vykreslenie  
bodov, alpha určuje priehľadnosť  
  
plt.plot([y_test.min(), y_test.max()], [y_test.min(), y_test.max()],  
'r--') # Referenčná čiara ideálnych predikcií  
  
plt.xlabel("Skutočné hodnoty (y_test)") # Popis osi x  
  
plt.ylabel("Predikované hodnoty (y_pred)") # Popis osi y  
  
plt.title("Skutočné vs. Predikované hodnoty") # Nadpis grafu  
  
plt.grid(True) # Zobrazenie mriežky pre lepšiu čitateľnosť  
  
plt.show() # Vykreslenie grafu
```

Interpretácia grafu

Ideálny model by mal mať všetky body ležiace na červenej čiarke ($y_{pred} = y_{test}$). Čím bližšie sú body k tejto čiarke, tým lepší je model. Rozptyl bodov okolo čiary indikuje chybu modelu.

Poznámky k analýze:

- Body **nad čiarou** znamenajú, že model **nadhodnocuje** skutočné hodnoty
- Body **pod čiarou** znamenajú, že model **podhodnocuje** skutočné hodnoty
- Rovnomerný rozptyl bodov naznačuje konzistentný výkon modelu napriek rôznymi hodnotami
- Zhluky bodov ďaleko od čiary môžu indikovať problematické oblasti v dátach



Čo sa naučíme?

1. Čo je viacnásobná regresia a kedy sa používa?
2. Ako sa líši viacnásobná regresia od jednoduchej regresie?
3. Ako sa vyberajú vhodné vstupné premenné do modelu?
4. Ako škálovať vstupné premenné a prečo je to dôležité?
5. Ako sa vyhodnocuje výkon modelu s viacerými premennými?
6. Na čo si dať pozor pri viacnásobnej regresii?
7. Profit

Zhrnutie kurzu



Základy strojového učenia

Pochopenie základných konceptov strojového učenia, rozdielu medzi regresiou a klasifikáciou, a prípravy dát pre modelovanie. Študenti sa naučia o učení s dohľadom a bez dohľadu, dátovej analýze, vizualizácii údajov a o dôležitých metrikách pre hodnotenie modelov. Tiež sa oboznámia s princípmi rozdelenia dát na trénovacie a testovacie množiny.

2

Lineárna regresia

Implementácia jednoduchej a viacnásobnej lineárnej regresie, interpretácia koeficientov a vyhodnocovanie modelov. Študenti sa naučia odvodiť rovnicu regresnej priamky, pochopiť význam koeficientu determinácie (R^2), strednej kvadratickej chyby (MSE) a analyzovať reziduály. Tiež sa oboznámia s predpokladmi lineárnej regresie a diagnostikou modelu.

3

Výber premenných

Techniky na identifikáciu a výber najdôležitejších premenných pre regresné modely. Študenti sa naučia používať metódy ako dopredná a spätná selekcia, rekurzívna eliminácia príznakov, a výber založený na penalizácii. Oboznámia sa aj s technikami na analýzu multikolinearity pomocou faktora inflácie rozptylu (VIF) a pochopia dôležitosť balansovaného prístupu k výberu premenných.



Regularizácia

Použitie Ridge, Lasso a ElasticNet regresie na zlepšenie generalizácie modelov a riešenie multikolinearity. Študenti pochopia matematické princípy L1 a L2 regularizácie, naučia sa správne kalibrovať hyperparameter alpha pomocou krízovej validácie, a získajú praktické skúsenosti s implementáciou týchto techník v Pythone využitím knižnice scikit-learn. Tiež sa oboznámia s interpretáciou modelov po regularizácii.

Machine Learning

Course Curriculum

